LFG-DOT: A Hybrid Architecture for Robust MT

Andrew Way,
Department of Language & Linguistics,
University of Essex,
Colchester, UK.

Thesis submitted in fulfilment of the requirements for
the degree of Doctor of Philosophy

March 21, 2001
Contents

1  Introduction to Machine Translation .............................. 10
   1.1  Background .................................................. 10
   1.2  Paradigmatic Approaches to MT ................................ 11
      1.2.1  Rule-based Approaches .................................. 11
             Advantages and Disadvantages of Rule-Based MT Systems 14
      1.2.2  Statistical Approaches .................................. 16
             The Language Model ....................................... 17
             The Translation Model .................................... 17
             An Example ................................................ 18
             Advantages and Disadvantages of Statistical MT Systems 19
      1.2.3  Example-based Methods .................................. 20
             Advantages and Disadvantages of EBMT ................... 21
      1.2.4  Hybrid Approaches ...................................... 23
   1.3  Current Impediments to Progress in MT ...................... 24
      1.3.1  Ambiguity ............................................... 24
      1.3.2  The Subset Problem ..................................... 25
      1.3.3  The Combination of Exceptions Problem ................ 29
   1.4  Summary of Thesis ........................................... 32

2  The DOP Approach to Language Processing .................... 34
   2.1  A Comparison of DOP with other Probabilistic Techniques .. 44
      2.1.1  Why Statistics? ........................................... 44
      2.1.2  Why DOP? .................................................. 47
   2.2  Other DOP models: DOP2 and Beyond .......................... 56
      2.2.1  DOP2: Dealing with Unknown Words ..................... 56

1
2.2.2 DOP3: an Adaptive Dynamic System .......................... 57
   The Good-Turing Method .................................... 57
   Experiments with DOP3 ........................................ 60
2.2.3 Some General Properties of DOP Models ........................ 60
   Probability of Sentences ..................................... 61
   Calculating the Probability of NP Subjects in DOP .............. 61
   Properties of Redundancy in DOP ................................ 67

3 Translation using LFG .............................................. 70
  3.1 On the Nature of Representations and Constraints ............... 73
  3.2 A Problematic Case of Translation for LFG: Headswitching .... 74
     3.2.1 Embedded cases of Headswitching ........................ 74
     3.2.2 Scoping of Multiple Adverbs .............................. 79
        Default vs. Specific Translations ........................... 81
        Translation using Restriction .............................. 83
  3.2.3 NPs containing Modifiers .................................... 85
     Linear Logic .................................................. 92
     Underspecified Semantic Representations ....................... 96
  3.3 Other Translation Problems .................................... 98
     3.3.1 Multiple Members of the Adjunct Set ...................... 98
     3.3.2 Anaphoric Dependencies ................................. 101
  3.4 Generation in LFG .............................................. 105
     3.4.1 Structure-Driven Generation .............................. 105
        Generation from F-structures .............................. 105
        Generation from S-structures .............................. 107
     3.4.2 Generation from Under- and Overspecified Structures .... 108
     3.4.3 Ambiguity-preserving Generation ......................... 109
     3.4.4 Discussion ............................................... 109
  3.5 Summary ........................................................ 109

4 LFG-DOP: A New, Hybrid Architecture for NLP ....................... 111
  4.1 Opportunities for Hybridity: LFG-DOP .......................... 111
     4.1.1 The Original LFG-DOP Model ............................. 111
        Observations on Corpus Size .............................. 125
4.1.2 A New LFG-DOP Model ................................. 129
4.1.3 Alignment Problems in LFG-DOP . ....................... 131
4.2 DOP adds Robustness to LFG .............................. 132
  4.2.1 Other Methods of Disambiguating LFG Analyses .... 136
    Ranking LFG Analyses .................................. 137
    Optimality Theory Constraint Ranking ..................... 138
    Optimality-theoretic LFG ................................ 140
    Comparing these Approaches ............................... 140
4.3 Restriction and Discard .................................. 141
4.4 The Discard Function in LFG-DOP ....................... 142
  4.4.1 Limiting the Number of Fragments .................... 142
4.5 A Reinterpretation of the Discard Function ................ 143
  4.5.1 The Role of Discard in each Stage of the Translation Process 145
  4.5.2 A Summary of the Two Interpretations of the Discard Function 146
4.6 Improving the Efficiency of DOP-based models .......... 146
  4.6.1 LFG-DOP and Efficiency ............................... 151
4.7 Semi-Automatic Generation of LFG-DOP Corpora ........ 152
  4.7.1 Grammar Compaction .................................. 156
  4.7.2 Enabling Automatic Annotation ......................... 159

5 Applications of DOP in Translation ........................ 162
  5.1 DOP as an Exemplar of Translation Memory ............. 162
  5.2 Data-Oriented Translation (DOT) ......................... 163
    5.2.1 DOT1 ............................................. 163
    5.2.2 Some Limitations of DOT1 ......................... 169
    5.2.3 DOT2: An Improved Model of Translation .......... 173
      Translations in Context ................................ 175
      Testing Translation Candidates in DOT2 ................. 179
      DOT2 and some ‘hard’ Translation Cases ................. 183
  5.3 Outstanding Issues ..................................... 186

6 LFG-DOT: A New, Hybrid Model of Translation ............ 189
  6.1 Introduction ........................................... 189
  6.2 Possible Models for LFG-DOT ............................ 190
6.2.1 LFG-DOT Model 1: Translation via $\tau$ ................................. 191
Summary .................................................. 197
6.2.2 LFG-DOT Model 2: Translation via $\tau$ and $\gamma$ ..................... 197
Using DOP Techniques on a ‘Treebank’ of F-structures .................. 202
Comparing LFG-DOT2 with DOT Models ................................. 204
Summary .................................................. 209
6.2.3 LFG-DOT Model 3: Translation via $\gamma$ with Monolingual Filtering ............ 210
Can LFG-DOT3 Improve on LFG-MT and CAT? ......................... 211
Can the $\tau$ function be reconstructed? ................................ 218
Is C-structure the Right Level at which to state the Translation Relation? ....... 221
Summary .................................................. 222
6.2.4 LFG-DOT Model 4: Translation via $\gamma$ and ‘Extended Transfer’ ............. 223
Summary .................................................. 228
6.3 Probability Models for LFG-DOT ........................................... 229
6.3.1 Model 1: Translation via $\tau$ ......................................... 229
6.3.2 Model 2: Translation via $\tau$ and $\gamma$ ................................ 232
6.3.3 Model 3: Translation via $\gamma$ with Monolingual Filtering ............. 234
6.3.4 Model 4: Translation via $\gamma$ and ‘Extended Transfer’ ................. 234
6.3.5 Other Possible Probability Models ..................................... 234
Random Fields ........................................... 235
Cormons’ Approach ......................................... 238
Probability in Terms of Occurrence in Derivations ...................... 239
6.3.6 Incorporating Discard into the LFG-DOT Probability Models ........... 242
Impact on Corpus Size in LFG-DOT1 .................................... 242
Impact on Corpus Size in LFG-DOT2 .................................... 245
Impact of Lexicalization on Corpus Size in LFG-DOT1 ..................... 246
Allocating a Fixed Probability Mass to Discard Fragments .............. 248
6.3.7 What Probability Model do we need? ................................... 249
6.3.8 What would be involved in building an LFG-DOT MT System? ............ 250

7 Conclusions 252
# List of Figures

1.1 The Vauquois Pyramid ............................................. 12
1.2 The Subset Problem ............................................. 27
2.1 The complete Tree-DOP multiset of fragments for the sentence *John swims* .......................... 35
2.2 Derivations for the sentence *John swims* ................................ 37
2.3 The complete set of fragments derivable from the sentences *John swims* and *Peter laughs* .......................... 40
2.4 Possible derivations of *Peter swims* given the Treebank in Figure 2.3 .......................... 42
3.1 Interim Derived Quadruple using F-structure Driven Generation .................. 106
4.1 The complete LFG-DOP treebank (no Discard) for the sentence *John swims* ............... 114
4.2 The additional LFG-DOP fragments via Discard for the sentence *John swims* .................. 122
4.3 The LFG-DOP treebank for the sentence *John fell* ........................................ 133
5.1 The complete DOT treebank for the linked sentence pair ⟨*John swims, Jan zwemt* ⟩ ........... 163
5.2 The monolingual DOT treebanks for the sentences *John likes Mary* and *Marie plait à Jean* .......................... 165
5.3 The complete DOT treebank for the *like* ⟷ *plaire* case ........................................ 166
5.4 DOT Derivations for *John commits suicide* ⟷ *Jean se suicide* ................................ 182
5.5 The monolingual DOT VP-fragments for a treebank built from the German examples in (343) 185

6.1 Sample LFG-DOP derivations for *John commits suicide* ........................................ 196
6.2 The complete LFG-DOT2 treebank (no Discard) for *John swims* ⟷ *Jan zwemt* ............... 199
6.3 LFG-DOT2: Translating using Different Levels of Linguistic Representation .................. 200
6.4 The linked F-structure fragments derivable via Discard for *John swims* ⟷ *Jan zwemt* ........ 204
6.5 The set of aligned F-structure pairs (no Discard) for *John swims* ⟷ *Jan zwemt* ............... 205
6.6 LFG-DOT2 derivations for *John commits suicide* ⟷ *Jean se suicide* .................. 206
6.7 LFG-DOT3 representation for *I think that the baby just fell* ⟷ *Je pense que le bébé vient de tomber* .......................... 212
6.8 Simplified LFG-DOT3 representation for *Le gouvernement sait permettre que les étudiants le fassent* \(\longrightarrow\) *The government knows how to permit the students to do it* . . . . . . . . . . . . 215

6.9 Simplified LFG-DOT3 representation for *On sait le faire* \(\longrightarrow\) *It is known how to do it* . . . . 217
List of Tables

2.1 Results of De Pauw’s (2000) experiments using Pattern-Matching .......................... 49
2.2 Adjusted Frequencies for NP-fragments from the ATIS corpus .............................. 58
2.3 P(S) \neq P(NP=SUBJ).P(VP) ........................................................................... 62

4.1 Number of LFG-DOP Fragments for John swims (no Discard) ............................. 125
4.2 The Effect of Discard on the Number of Fragments produced ............................. 126
4.3 The Explosion of LFG-DOP fragments in a corpus of 440 sentences ..................... 127
4.4 Results of ‘reparsing’ the AP Treebank to produce F-structures ......................... 155
4.5 Compacting the Rules derived from a Treebank using ‘Supertags’ .......................... 158
4.6 Explosion of F-structures in a Compacted Grammar ........................................... 159
4.7 LFG’s Grammaticality Checks act as a Filter on Combinatorial Explosion of F-structures ................................. 159
4.8 Precision & Recall for Automatic Annotation of F-structures using Templates ........ 161

5.1 Number of Fragments in Two Monolingual Treebanks .......................................... 180

6.1 Relative Frequencies and Probabilities of the Trees in (423) ................................. 241
6.2 Relative Frequencies and Probabilities of the Trees in (425) ................................. 242
6.3 The Effect of Discard on Monolingual LFG-DOP Treebanks ............................... 243
6.4 Detailed Breakdown of the Discard Fragments in a French LFG-DOP Treebank .... 243
6.5 The Effect of Discard on LFG-DOT Treebanks .................................................. 245
6.6 The Impact of Lexicalization on Treebank Size ................................................... 246
6.7 The Effect of the Discard operation on the Verbmobil and Homecentre corpora .. 248
Acknowledgements

Much of this work was done while at the CLMT group at the University of Essex. Special thanks to Doug Arnold, my supervisor, for many hours of fruitful discussion during those, and subsequent times. I should also thank Doug (along with Jacques Durand) for giving me my first job in NLP on the Eurotra(vel) project.

Thanks are also due to Louisa Sadler for introducing me to LFG in the first place, as well as to Frederik Fouvy and Louisa for further useful discussion while I was in Essex. In addition, thanks are due to Rens Bod, Joan Bresnan, Boris Cormons, Anette Frank, Ron Kaplan, Arjen Poutsma, Khalil Sima’an, Josef Van Genabith, Menno van Zaanen and Tony Veale for many helpful comments. Parts of this work were also presented in journals and at various conferences: TMI, Machine Translation, JETAI, AICS-99, NODALIDA-99, LFG-99, ATALA-Treebanks, EACL-99, TAG+5, CLIN-00—thanks also to the reviewers of those papers, as well as to the reviewers of unsuccessful submissions! This thesis has also benefitted considerably from the many astute and incisive comments proposed by my examiners, Louisa Sadler and Rens Bod, so I am very grateful to them. Any remaining errors or omissions are of course my fault.

Thanks to Dublin City University for funding this research, and colleagues in the School of Computer Applications for covering for me during my periods of enforced absence.

Finally, special thanks are due to my parents for their support during my academic career, and to Julie for her boundless, unstinting patience, love and understanding.
Abstract

This thesis explores the possibility of combining Data-Oriented Parsing (DOP) with the conventional transfer rules of Lexical Functional Grammar (LFG) to derive a new model for Machine Translation (MT).

We begin by describing why none of the main paradigmatic approaches to MT currently suffice to the standard required. Nevertheless, each of these approaches contains elements which if properly harnessed should lead to an overall improvement in translation performance. It is in this new hybrid spirit that our search for a better solution to the problems of MT can be seen.

We summarize the original DOP model (Bod 1992), and compare it to other statistical models of language. We also introduce LFG, and describe how the correspondence-based model of translation (LFG-MT: Kaplan et al., 1989) works. We describe a range of phenomena which poses considerable problems for LFG-MT. We also report on previous attempts at solving these problematic constructions using LFG, almost all of which introduce further problems.

We then describe the hybrid LFG-DOP model of (Bod & Kaplan, 1998). LFG analyses can be automatically ranked using this model, and well-formed analyses can be generalized via Discard to allow ill-formed and previously unseen strings to be handled. Such data cannot be handled at all in LFG. We also describe previous efforts at solving the search problem in DOP-based approaches, and present novel methods of restricting the number of fragments derived in LFG-DOP models. In addition, we report on attempts at semi-automatically deriving LFG-DOP corpora from existing treebanks, which we hope will permit the large-scale testing of the LFG-DOT models presented here.

We describe the DOT models of translation (Poutsma 1998; 2000) based on DOP. We demonstrate that DOT1 is not guaranteed to produce the correct translation, despite provably deriving the most probable translation. We also describe the DOT2 translation model, which solves most of the problems of DOT1. Notwithstanding the success of DOT2, any system based purely on trees will ultimately be found wanting as a general solution to the wide diversity of translation problems, as certain linguistic phenomena require a description at levels deeper than surface syntax.

We then show how LFG-DOP can be extended to serve as a novel hybrid model for MT, LFG-DOT, which promises to improve upon DOT, LFG-MT and other tree-based MT systems. We conclude with the contributions of the thesis, and some questions for further research.
Chapter 1

Introduction to Machine Translation

1.1 Background

A welcome recent development in the field of MT is the recognition that none of the four main paradigmatic approaches to MT, namely:

- Transfer-based, e.g. META, (Bennett & Slocum, 1985); Eurotra, (Arnold & des Tombe, 1987)
- Interlingua-based, e.g. Rosetta, (Landsbergen, 1989; Rosetta, 1994); KBMT, (Goodman & Nirenburg, 1991)
- Statistics-based, e.g. Candide, (Brown et al., 1990; 1992a)
- Example-based, e.g. Gaijin, (Veale & Way, 1997); EDGAR, (Carl & Hansen, 1999)

in themselves perform the task of fully automatic, high quality translation to the standard required. This description of the status quo differs quite considerably from the rather more passionately held views in the less recent past in this area. In the late 70's and early 80's, it is fair to say that the MT protagonists were either in the transfer or interlingua camp, and exchanges between the various proponents of either type of system were often rather heated. Then, in the late 80's the first purely statistical approaches to MT arrived, so the camps diversified into the adherents of the data-driven approaches of statistics- and example-based MT on the one hand, and the supporters of the rule- and constraint-based approaches of transfer and interlingual systems on the other. We will summarize some of the main advantages and disadvantages of each basic system type in the next few sections.

More recently, however, more and more researchers in MT have advocated a hybrid view of solutions to the problems of translation. It is now accepted that these paradigmatic approaches do not yet work well enough on their own. Furthermore, many researchers in MT believe that (components from) these basic types might be combined in models to improve the overall translation quality. It is in this new hybrid spirit that our search for a better solution to the problems of MT can be seen. We propose a combination of the syntactic

1.2 Paradigmatic Approaches to MT

The next sections provide a discussion of the four main paradigmatic approaches to MT: the two rule-based methodologies—transfer and interlingua—and the two empirical, data-driven techniques—statistical MT and example-based MT.

1.2.1 Rule-based Approaches

Most, if not all textbooks on MT divide up their discussion on rule-based MT (RBMT) into sections on transfer-based methods and interlingual methods. There are indeed differences between the two approaches. The transfer methodology essentially involves relating two (or more) languages at the level of syntax. Accordingly, therefore, the source and target intermediate representations reflect the structural relations found in the source and target strings. Where the structural representations differ to any great degree between the two languages, the task of transfer may be complicated: although the main aim of the transfer approach is the facilitation of ‘simple transfer’ (essentially, word for word substitution between source and target languages at the level of intermediate representation), this is often unachievable. In this situation, complex rules need to be written to map cases of ‘structural transfer’ between the languages involved.

Such cases of ‘complex transfer’ can often be overcome in interlingual systems by analysing translation problems to a ‘deeper’, semantic level of representation. One such example is the complex transfer case of relation-changing verbs, as in (1):

\[
\text{EN: You like her} \quad \leftrightarrow \quad \text{ES: Ella te gusta} \\
\text{DE: Sie gefällt Dir} \\
\text{FR: Elle te plait}
\]

Possible intermediate representations for the examples in (1) appear in (2):

Here, the subject in English gets a dative realization in Spanish, French and German, while the English object becomes the subject in the other three languages. Note that transfer between Spanish, German and French in (1) is simple: it is the task of the respective generation components to take the rightmost object in (2) and produce the appropriate strings in (1). Note that these intermediate representations conform to a canonical order, which abstracts away from the differences seen at the level of surface structure.
Hutchins & Somers (1992:116) demonstrate that the differences between the languages at the level of surface structure can be neutralized by using a case-based representation, as in (3):

\[
\begin{align*}
\{\text{lex}=\text{like}/\text{plaire}/ \\
\text{gefallen}/\text{gustar}, \\
\text{cat}=v, \text{tense}=\text{pres}\}
\end{align*}
\]

(3)

\[
\begin{array}{ll}
\{\text{theta}=\text{experiencer}, \\
\text{cat}=\text{pers.pron}, \\
pers=2, \text{num}=\text{sg}\} & \{\text{theta}=\text{patient}, \\
\text{cat}=\text{pers.pron}, \\
pers=3, \text{num}=\text{sg}, \text{gender}=\text{fem}\}
\end{array}
\]

We can see that the idiosyncrasies of the strings in (1) (and the structures in (2)) have been mapped into a neutral representation in (3): \textit{you} is a subject pronoun in English, \textit{te} is a direct object preceding the verb in French and Spanish, and \textit{Dir} is a dative pronoun following the verb in German—all are mapped to the theta role of \textit{experiencer} in (3). Similarly, \textit{her} is an object pronoun in English in (1), whereas \textit{ella}, \textit{sie} and \textit{elle} are subject pronouns in Spanish, German and French respectively: all are mapped to the \textit{patient} semantic role in (3).

Of course, it is easier to conceive of a transfer-based analysis component which can parse strings such as those in (1) into the sort of representations in (2), than it is to imagine an analysis component in an interlingual system capable of transforming the input strings into the case-based structures in (3). The same holds for the respective generation components. That is, neutralizing the cases of complex transfer in a transfer system does not come cost-free: the additional overhead moves to the analysis and generation components in an interlingual MT system.
This can be seen quite clearly from Figure 1.1, the ‘Vauquois Pyramid’. This diagram is familiar to all MT researchers, and shows the relative effort involved for each component in the translation process. One can see that in direct systems, very little analysis and generation is performed at all. Given that such systems are by and large neither rule- nor statistics-based, relying as they do on their huge comparative lexicons, they will not be discussed further here. If truly language-neutral representations were possible, then such an interlingual system would be situated at the peak of the pyramid; that is, the work involved in such a system would befall the analysis and generation components only. Any intermediate representations would be neutral with respect to all languages. We have chosen to represent a transfer system in Figure 1.1: the length of the arrows represents in a schematic manner the relative workload of each component in the MT system. Most versions of the Vauquois Pyramid draw the transfer arrow horizontally, whereas we have chosen to portray this at an incline. This is intended to represent the fact that the task of analysis in most transfer systems is usually more onerous than generation. For instance, while most MT systems can expect to have to parse both active and passive strings, the system designers can choose to output just active (or passive) ones in generation. That is, faced with John kissed Mary and Mary was kissed by John, a transfer-based MT system might represent these strings using the respective intermediate representations in (4):

\[
\begin{align*}
&\{\text{lex=\textit{kiss}}, \\
&\text{diath=active,} \\
&\text{cat=\textit{v},} \\
&\text{tense=past}\} & & \{\text{lex=\textit{kiss}}, \\
&\text{diath=passive,} \\
&\text{cat=\textit{v},} \\
&\text{tense=past}\} \\
&\{\text{lex=\textit{john}}, \\
&\text{cat=\textit{np},} \\
&\text{role=agent}\} & & \{\text{lex=\textit{mary}}, \\
&\text{cat=\textit{np},} \\
&\text{role=patient}\} & & \{\text{lex=\textit{john}}, \\
&\text{cat=\textit{np},} \\
&\text{role=agent}\} & & \{\text{lex=\textit{mary}}, \\
&\text{cat=\textit{np},} \\
&\text{role=patient}\}
\end{align*}
\]

Assuming that active and passive strings are found in one’s corpus, the MT system is forced to confront examples of both types, unless translation is to fail. If representations such as (4) are input into the generation phase, however, a processing decision might be taken to produce only active (or passive) strings from either intermediate representation, but not both. While translation theorists might baulk at the idea that a possible translation of Jean a embrassé Marie is Mary was kissed by John,1 at the level of engineering, a system designer might be delighted that such a ‘translation’ is obtained: it is a well-formed sentence, and at one level can be said to encode the event being described. What is more, one’s generation component requires fewer rules if passives are not produced, so that the system runs faster.

Notwithstanding these obvious differences, when it comes to comparing the transfer and interlingual approaches to the statistical methodologies, they can quite simply be grouped together as exemplars of the rule-based approach to MT. Indeed, many of these perceived differences often disappear in practice: so-called transfer systems which embody a rich theory of semantic features may have more similarities with interlingual systems than other transfer-based systems, while given the impossibility of mapping lexical items to a language-neutral representation, many interlingua-based systems interpose a bilingual dictionary between their analysis and generation components, similar to transfer systems.

---

1Here, and in many future examples, we ‘translate’ proper names purely in order to differentiate completely source and target representations and strings.
Advantages and Disadvantages of Rule-Based MT Systems

The biggest single problem with rule-based (primarily transfer) systems is that of knowledge acquisition. This takes several forms (Su & Chang, 1992:255):

- Wide coverage of texts is difficult to achieve. Knowledge is (normally) restricted to (theoretically interesting) interactions of linguistic phenomena (whether these occur frequently or not in real corpora is deemed irrelevant). Consequently expansion from dealing with ‘toy’ grammars and lexica often leads to lack of robustness.

- Given that the translation data is (often) invented (rather than extracted from real corpora), it is difficult to maintain consistency in the knowledge bases between developers.

- Approaches are (often) based on existing linguistic theories, which are themselves incomplete. Therefore, it is tempting to resort to ad hoc procedures when faced with constructions not dealt with in the theory.

- They find it hard to deal with ill-formed input. Again, given the fact that they are often based on grammatical theories, most only accept well-formed strings. Real text, unfortunately, is not always so accommodating.

Other criticisms having similar roots include:

- Expanding one’s coverage may cause newly added rules to impinge on others in an unpredictable fashion, causing previously correct behaviour to be inadvertently undone. This might be termed a problem of tuning.

- There is (normally) no systematic basis to the acquisition of rules, so that while being of theoretical interest, such systems may be of little real relevance.

- It is difficult to handle uncertainty, i.e. if such systems incorporate a preference mechanism, this (normally) has no empirical objectivity or consistency underpinning it.

Given this, one might question why the transfer paradigm has proved the most popular over the years with MT developers. Of course, there are a number of advantages to transfer-based approaches, although many of these so-called advantages are in fact ‘non-disadvantages’ inherent in other approaches, notably interlingua-based systems.

For the purposes of this discussion we will assume the following applies equally to interlingual and knowledge-based MT (KBMT). Such approaches are viewed as attractive in theory, but unattainable in practice. Truly language neutral interlinguae are unachievable, and indeed, for closely related languages (such as Spanish and Portuguese, for instance) it makes little sense to ignore their similarities in translation. Consequently, a more pragmatic approach is usually taken which stops shy of such an intellectually appealing, but ultimately impractical stipulation that the intermediate representation be language-neutral, accepting instead that they be merely language-independent.\(^2\)

\(^2\)Compare the sentences:
Nirenburg et al. (1992:51) term this the ‘maximalist’ view of interlingua, and discuss it among a plethora of oft-cited criticisms of such approaches, focusing particularly on arguments for and against ‘meaning-oriented’ MT. One of the criticisms (op cit., p.43) is that meaning is not required for translation, a view with which advocates of statistics-based methods would agree (e.g. Brown et al., 1990, 1992a). Nirenburg et al. (op cit., p.46) state that the processes involved in the statistical approach, particularly ‘viewing language not as a productive system but as a fixed set of canned locutions ... moves MT out of applied science and into pure engineering’; not that there is anything wrong with this per se, of course. They continue:

‘Completely uninterpreted comparison (of text corpora) will lead to errors simply because the human translators who produced the translations in the corpus in the first place do not translate word-for-word or even sentence-for-sentence.’ (Nirenburg et al. op cit., p.46)

While there is little doubt that this is true, we intend to show that a hybrid approach combining a DOP treebank\(^3\) with the linguistic structures provided by LFG ought to produce fewer such errors, leading to an overall improvement in translation output, both in terms of quality and robustness. In keeping with most (if not all) MT systems, which do not consider context outside the bounds of one sentence, we restrict our investigation to intra-sentential translational phenomena. In addition, unlike much of the literature on MT systems, we investigate to what degree combinations of complex phenomena are handled by a number of MT systems, including the new LFG-DOT systems proposed in this thesis.

One of the major criticisms of RBMT concerns the generation of the target language, where transfer-based approaches tend to preserve the syntactic structure of the source text in translation, thereby leading to less than optimal translations. The issue of the influence of source tree structures on the target structures built has been discussed at length in the MT literature. For instance, Nirenburg et al. point out (op cit., p.55):

‘Direct structural correspondences between certain pairs of languages can be exploited in MT systems of a particular type, but they should be treated as idiosyncratic occasions rather than phenomena that occur as a rule ... However, if an MT system does not possess sufficient knowledge to analyze source language texts deeply enough ... it may rely on preserving the syntax of the source text in the target text as a very crude default heuristic’.

Except for simple translation cases where isomorphic structures in both source and target are to be expected, human translators use structure-preserving translations only as a last resort. While such translations may be grammatically correct, they are usually poor choices from a stylistic point of view. In MT systems,

\(\textbullet \) John likes to swim.

\(\textbullet \) Jan schwimmt gerne.

In the English sentence, we see that \textit{like} is the main verb, with \textit{swim} occurring in the complement clause. In German, we see that \textit{schwimmen} is the main verb, with the ‘liking’ element portrayed by the adverb \textit{gerne}. \textit{Like} ---\textit{gerne} can be handled interlingually—but in truly interlingual systems it must be handled \textit{neutrally}. This could be with \textit{like} or \textit{schwimmen} as head: the first option would mirror the English, whereas the second would be more like the German, but neither would be neutral.

\(^3\)We draw the reader’s attention to the distinction between a DOP (and LFG-DOP) treebank, which consists of a database of tree fragments (and associated LFG f-structure fragments in LFG-DOP), and a treebank such as the Penn Treebank (Marcus et al., 1993), which might serve as the input into the DOP fragmentation process from which the DOP (or LFG-DOP) treebank is derived.

15
however, the issue is much harder to avoid. Indeed, in some cases it is the default option to cut down on the computational complexity required.

Such criticisms have been taken into account in systems as diverse as Shake & Bake MT (Whitelock 1992; Beaven 1992) and Statistical MT (Brown et al., 1990; 1992a), where the target strings are produced from a number of target words in a ‘bag’ according solely to rules in the target grammar, and with no reference to the source string at all. Similarly, this problem of structure-preserving translation may be avoided in EBMT (cf. section 1.2.3): since many EBMT systems do not encode any structural representation of the string pairs, the structure of the source string cannot by definition be imposed on the target equivalent.

We shall show that in dealing with the like $\rightarrow$ plaire case in (1), for instance, the DOT1 model of translation has no choice but to impose the source structure on the target given the strict interpretation of the DOP substitution operation, resulting in wrong translations for this, and indeed all cases where the word order differs between languages. DOT2, meanwhile, uses a different composition operator and enables these translation cases to be handled correctly. Our proposed models for LFG-DOP MT also avoid this criticism. Target strings are either produced from target LFG t-structures (LFGDOT1 and 2), or by synchronously constructing target PS-trees while parsing the source representation (LFGDOT2-4): source trees are linked only to bona fide target structures, so that nothing can be built which is not a well-formed representation according to the target ‘grammar’, i.e. those tree fragments already in the treebank. Consequently all LFG-DOT representations have a clear notion of ‘grammaticality with respect to the corpus’.

1.2.2 Statistical Approaches

After a relatively long period in the wilderness, empirical methods have recently made a comeback in many areas of NLP. As pointed out in (Dagan & Church, 1994), some of these include speech synthesis (e.g. Sproat et al., 1992), question-answering systems (e.g. Kupiec, 1993), terminology research (e.g. Isabelle, 1992), information retrieval (e.g. Landauer & Littman, 1990), bilingual lexicography (e.g. Smajda, 1992), and word-sense disambiguation (e.g. Brown et al., 1991). Empirical approaches to MT have also recently become more popular primarily for two reasons:

- the (relative) failure of rule-based approaches;
- the increasing availability of machine-readable text.

The first of these requires little further comment; if rule-based approaches have not brought the success anticipated then alternative methodologies need to be tried, especially where they have proven their worth in other areas of NLP. Where rule-based methods are still being used, attempts are being made to increase their success by embedding them in controlled language applications (cf. Adriaens, 1996).

Secondly, given the hundreds of millions of megabytes of machine-readable text available today, together with the speed of modern day computers, the statistics necessary for empirical approaches to the problems of MT can readily be gathered. Indeed, experiments have been performed using the entire World Wide Web (e.g. Grefenstette, 1999) as a ‘virtual corpus’ in which statistics for a target language are used to disambiguate nominal compounds which are ambiguous in the source. Given that the web is likely to be
the largest ‘corpus’ in existence for any language, its use in tasks like this can overcome data sparseness and produce quite reasonable results.

In the context of MT, ‘pure’ approaches using statistics are diametrically opposed to the rule-based approach, in that rather than expressing formal linguistic information in an explicit way, they rely wholly on probabilistic techniques. Given this, one might take the extreme view (cf. Brown et al. 1990), that “every sentence in one language is a possible translation of any sentence in the other”.

The Language Model

What is needed is a language model which assigns probabilities to sentences, \( P(S) \) for source strings and \( P(T) \) for sentences in the target language. \( P(S) \) can be understood as the probability of the string \( S \) occurring. That is, given a string of words \( s_1 s_2 ... s_n \), we can write (5):

\[
P(s_1 s_2 ... s_n) = P(s_1) P(s_2|s_1) ... P(s_n|s_1 s_2 ... s_{n-1})
\]

As Brown et al. state (op cit., p.80), this “recasts the language modelling problem as one of computing the probability of a single word given all of the words which precede it in a sentence”. As an example, the probability of have and has occurring in a text might be very similar, but the probability of has occurring after he, she or it would be a lot higher than have occurring. One problem is that in a sentence of any length, the number of conditional probabilities would be very high. Consequently, most approaches of this type normally take into account only the preceding one (bigram models) or two words (trigram models) in making these calculations.

The Translation Model

The next stage is to develop a translation model, \( P(T|S) \), i.e. the probability that the source string \( S \) translates as the target string \( T \) in a given text. For simple sentences this probability will be high, as in (6):

\[
(6) \quad \text{John loves Mary} \rightarrow \text{Jean aime Marie.}
\]

Such strings which translate word for word are aligned exactly with one another, i.e. John is aligned with Jean, loves with aime, and Mary with Marie. Of course, this is not always the case, as shown in (7):

\[
(7) \quad \text{John does not love Mary} \rightarrow \text{Jean n’aime pas Marie.}
\]

Here we have the same word for word alignments as before, plus the new alignments of not with ne pas, and does with \( \emptyset \). This shows that a word in a source text can be aligned with more than one word in the target text (and vice versa), or with nothing in the target text (and vice versa). The number of French words produced by an English word in an alignment gives that alignment its fertility.

As Brown et al. point out (op cit., p.81), it seems to be the case that when translating between English and French, words at the beginning and end of a sentence in one language tend to be aligned with their
counterparts in the other, but again, this need not be the case. Sometimes a French word will appear a long way from the English word from which it was produced. This effect is known as distortion. This will allow, for instance, adjectives to appear after their nouns (on the whole) in French, but before them in English.

An Example

The product of the probability that S translates as T and the probability of S, i.e. $P(S)P(T|S)$, gives us the probability that pairs of source-target strings will occur, i.e. $P(S,T)$. What we need to do then is find that string $S$ which maximizes $P(S)P(T|S)$, if translation is seen as the problem of finding the $S$ that is most probable given $T$. Using Bayes' theorem (which relates probabilities) we can write (8):

$$P(S \mid T) = \frac{P(S)P(T \mid S)}{P(T)}$$

What is now required is a method for searching for that $S$ among all the candidate strings which gives the greatest value for $P(S)$ and $P(T|S)$. Given the potential number of sentences available, this is non-trivial, but an adaptation of the Viterbi algorithm (cf. section 2.1.2) suffices for this task.

To put all of this into practice, let us try to compute the probability of the alignment (John does not love Mary | Jean n'aime pas Marie). In order to do this we must calculate:

1. the fertility probabilities for each word (i.e. how likely it is to be translated as 1, 2, ..., x words);
2. the translation (or word-pair) possibilities for each word in each language;
3. the set of distortion probabilities for each source and target position.

The first two listed here can be portrayed as in (9):

$$P(fertility = 1|John) \times P(\text{Jean}|John) \times$$
$$P(fertility = 0|\text{does}) \times$$
$$P(fertility = 2|\text{not}) \times (P(\text{ne}) \times P(\text{pas})) \times$$
$$P(\text{fertility} = 1|\text{love}) \times P(\text{aimer}|\text{love}) \times$$
$$P(\text{fertility} = 1|\text{Mary}) \times P(\text{Marie}|\text{Mary}).$$

Factoring in the distortion probabilities involves at its simplest the assumption (op cit., p.91) that the position of the target word depends only on the length of the target string and the position of the source word. Therefore a distortion probability has the form $P(i|j, l)$, where $i$ is a target position, $j$ a source position, and $l$ the length of the target sentence.

As with example-based MT, statistical methods rely on enormous volumes of good quality data being available in the two languages where translation is being performed. At this point in time, this is available only in very few instances, so the technique is currently limited in its application. The Canadian Hansards are the example bilingual corpus used by Brown et al. (1990) for their experimental work. Even where such
data is available, the results are not as impressive as one would like. Brown et al. (op cit., p.83) report a rate of 48% 'successful translation' (5% exact match with the reference translation, 25% 'alternate' (i.e. similar to the reference) translations, and 18% different translations (i.e. quite far removed from the reference translation, yet still regarded as acceptable translations) on a set of 73 sentences. One of the reasons for this relatively low performance is that morphologically related words are treated as different, so that the contribution of such words to distributional information about their morphological cousins is lost.

Consequently, in subsequent work (Brown et al., 1992a), they shy away from the 'pure' statistical approach in recognizing the need to incorporate low-level linguistic information in order to capture morphological variants of each word to improve their statistical model. This shows that there is a role for statistical information in the MT process, but rather than relying solely on such techniques, what remains is the question of how best to combine statistical and rule-based approaches.

Advantages and Disadvantages of Statistical MT Systems

Compared with rule-based approaches, 'pure' statistical methods assume no linguistic models, nor do they have 'any methods of strict well-formedness in mind' (Su & Chang, 1992:250). Consequently, it is possible to cope with ill-formed input using such approaches, while not being tied to any linguistic theory enables easy computation (cf. Brown et al. 1990). Other advantages of this type of approach include (Su & Chang 1992:252):

- Uncertainty is interpreted objectively;
- Consistency is maintainable even in large-scale systems;
- System parameters can be manipulated language-independently;
- Training is possible with little human intervention.

In addition, it might be argued that a good deal of the knowledge needed for MT is inductive, rather than deductive, in that while linguistics is induced from languages, no natural language is generated from any linguistic theory per se. All these facts would argue in favour of a statistics-based approach. However, there are problems too, as one might expect:

- The statistical approach requires huge, good quality bilingual (or multilingual) corpora to be available. This is currently the case for few languages only, rendering this approach to MT rather limited.
- If the corpora are too small, one faces the problem of sparse data, where one's statistical models could prove unreliable.
- They also need to be representative, for word frequency strongly depends on the domain and text type.

---

4 Nevertheless, the success rate of rule-based systems is not great either, and one has the further disadvantage of encoding all the necessary linguistic information - not even an issue with statistical approaches.

5 The question as to how ill-formed a text has to be before any processing of it is unmerited is merely noted here. Note also that it is possible (though unlikely) for such ill-formed data to skew the results obtained by such a system. The reader will note that this concern ultimately influences our proposals for models of LFG-DOP MT.
• Given the lack of any linguistic knowledge, the parameter space is normally impractically large. One tends, therefore, to sacrifice the quality of the statistical information (using bigram models rather than trigrams, for instance) at the expense of functionality (sparseness, again).

An additional criticism of statistical approaches to NLP is that almost all of them can deal only with local phenomena. That is, there are certain constructions (e.g. long-distance dependencies), which cannot be dealt with by most such methods. Furthermore, it would appear that the effect of distortion may cause the effectiveness of such an approach to be questioned between two languages whose surface orders are not closely mirrored (English and Japanese, say). Even between closely related languages, one can foresee problems: it is non-trivial to gather accurate statistics to find correlations between English and German verbs, for instance, which, in complex sentences at least, appear in rather different surface positions. We note here that these problems are less severe in DOP-based models than in other statistical approaches, and discuss this further in chapter 2.

1.2.3 Example-based Methods

Example-based MT (EBMT; also known as Memory-based, Case-based, Experience-based or Analogical Translation) is similar to statistics-based MT in that mapping rules from one language to another (or, in more complex cases, recursively between levels of representation of one language to (ultimately) levels of representation in the other) have no place. Instead, example translations are stored and matched against input text as candidate solutions to the problems of MT. What is involved then is to gather a bilingual corpus of paired translations and to find the best possible match among the set of candidate translations.

There are three stages to example-based translation:

1. matching fragments of the source text against the reference corpus;
2. identifying the corresponding translation fragments;
3. recombining these translation fragments into the appropriate target text.

EBMT is trivial if an exact match can be found in the database. More normally, however, we can expect to have to identify several relevant examples, each of which contain fragments which are ‘close’ matches to elements of the source string. Depending on the depth of alignment enabled in the EBMT system, the translation fragments corresponding to each of these close matches may be more easily discovered. These then have to be assembled to produce the translation. At a particular level of abstraction, it can be seen that each of these three stages correspond to the analysis, transfer and generation stages in RBMT. Indeed, Somers (1999) superimposes each of the stages on the Vauquois pyramid in Figure 1.1.

To give a very simple, straightforward example, let us attempt to translate John went to the baker’s into French. Let us assume that the EBMT matching stage finds the relevant examples in (10):
(10)  a. John went to school \textarrow{Jean est allé à l'école.} \\
    b. The butcher's is next to the baker's \textarrow{La boucherie est à côté de la boulangerie.}

From these examples, we might expect our EBMT system to isolate the useful translation fragments in (11):

(11)  a. John went to \textarrow{Jean est allé à} \\
    b. the baker's \textarrow{la boulangerie}

The final stage of the EBMT process would be to recombine the target fragments to produce the correct translation \textit{Jean est allé à la boulangerie}.

**Advantages and Disadvantages of EBMT**

One of the first examples of this approach was that of Sumita \textit{et al.} (1990), who list a number of advantages of EBMT over rule-based methods. These include (\textit{op cit.}, p.204):

- Computational Cost
- Improvement Cost
- System Building Cost
- Context-Sensitive Translation
- Robustness
- Reliability
- Context-independent System Knowledge (and consequently, Reusability)

One of the major advantages which is often claimed of EBMT is that the overall quality of translation increases incrementally as the set of stored translations increases.\textsuperscript{6} For example, Mima \textit{et al.} (1998) report that in their EBMT system, translation quality rose in an almost linear fashion, from 30\% with 100 examples to 65\% with all 774 examples. They also note that there seems to be a limit beyond which adding further examples does not improve the overall translation quality.

While the chances of finding an exact match become greater as the corpus size increases, there are two knock-on effects whose impact on the EBMT system should be minimized. Firstly, adding more examples has a computational cost, especially if the examples need to be parsed: some EBMT systems (e.g. Sato & Nagao, 1990) store examples as annotated trees, for instance. Whether this is the case or not, adding

\textsuperscript{6}Many software companies have reduced their translation costs considerably by using a related technique, namely translation memories (TM). In the localisation process, particularly between updates of software releases, a large part of the manuals remains exactly as in the original release. Matching techniques similar to those used in EBMT are employed to leverage material from the old legacy translations which is still relevant for the new software release. However, EBMT differs from TM in one important respect: EBMT is a technique for \textit{automatically} producing the 'best' translation from a set of stored examples, whereas TM is essentially a tool which presents best matches to the \textit{translator}, who is ultimately charged with producing the correct translation.

21
more examples causes more storage problems, and adds to the complexity of the search and retrieval stages of the EBMT process. It is, therefore, unclear that one of the purported advantages of EBMT, namely a lessening in computational cost, is indeed a real benefit. Secondly, adding more examples may not be useful in practice. For example, a newly added translation pair may be identical to, or overlap other examples. Where a system involves the computation of a ‘similarity metric’ (e.g. Somers et al., 1994), this may be influenced by the frequency of examples, so that the score attached to certain matches increases if a large number of similar examples are found. Alternatively, in other systems, identical or similar examples may just be redundant. Somers (1999) observes that “in such systems, the examples can be seen as surrogate ‘rules’, so that, just as in a traditional rule-based MT system, having multiple examples (rules) covering the same phenomenon leads to over-generation”.

In practice, therefore, EBMT does not proceed as straightforwardly as the examples in (10)–(11) might suggest. The two main problems for EBMT are boundary definition and boundary friction. The first of these describes the scenario where retrieved fragments may not be well-formed constituents. This is a particular problem for pure EBMT systems, where syntactic well-formedness needs to be ensured without grammatical information actually being employed. The second, boundary friction, is a problem in the retrieval process, in that context may not be taken into account. This may be illustrated by attempting to translate I have a big dog into German. An EBMT system might retrieve the close matches in (12):

(12) a. A big dog eats a lot of meat → Ein großer Hund frisst viel Fleisch.
   b. I have two ears → Ich habe zwei Ohren.

From these examples, we might expect our EBMT system to isolate the useful translation fragments in (13):

(13) a. A big dog → Ein großer Hund
   b. I have → Ich habe

In this case, these fragments would be combined to give the translation in (14), which is ungrammatical as we have an NP bearing nominative case in object position:

(14) * Ich habe ein großer Hund.

As to the other listed advantages, the cost of building an EBMT system from scratch is indeed less than building a rule-based system, especially where parsers and dictionaries for the languages in question are hard to come by. Even if aligned corpora are not available, these can be typed in by hand if necessary. Manual construction of aligned corpora, a sine qua non for such techniques, also avoids the considerable problem of trying to align source and target texts. Without outlining here the various strategies described in the literature, we simply state this to be a further problem for both statistical and example-based MT.

As for context-sensitive translation, all EBMT systems are corpus-based, and many corpora are sublanguage-based. The examples in the corpora are real examples, so that the EBMT system is required to deal only with the translational phenomena included in the sublanguage, and nothing else. This seems to contradict the last point of Sumita et al. (1990), namely that EBMT systems are reusable. While the examples themselves are of course reusable, like any MT system designed to handle a specific sublanguage, it is not readily portable
to another domain. Indeed, as Carl & Hansen (1999:623) point out, “the more an MT system is able to decompose and generalize the translation sentences, translate parts or single words of it and to re-compose it into a target language sentence, the broader is its coverage and the more it loses translation precision”. While this might be applicable to any type of MT system, in the context of their work it applies to EBMT, in which case one might require that all example-based translation be oriented towards a particular sublanguage, lest translation quality be compromised.

As with advocates of rule-based systems, there are extreme adherents who suggest that EBMT may be able to deal with the whole process of translation (Sato & Nagao, 1990; Veale & Way, 1997). There are other more moderate proposals which prefer instead to combine this approach with others (e.g. Sumita et al., 1990). Nowadays, the example-based approaches are not in conflict with the rule-based approach, so that it is widely believed that a mixture of different approaches could be of benefit. As Arnold et al. (1994:201) point out:

“What this suggests is that there is no radical incompatibility between example-based, and rule-based approaches, so that the real challenge lies in finding the best combination of techniques from each. Here one possibility is to use traditional rule-based transfer as a fall back, to be used only if there is no complete example-based translation”.

1.2.4 Hybrid Approaches

It should now be clear that dogmatic adherence to one methodology will result in sub-optimal results. Indeed, at least with reference to the claim that statistics-based techniques suffice, this is unnecessary since many regular aspects of language (such as subject-verb agreement, or agreement between a relative pronoun and a gap in a long-distance dependency, for instance) can be handled quite simply using rules.

There are a number of projects which use a statistical MT component in a multi-engine system. The most notable of these are Verbmobil (Wahlster, 1993), and the Pangloss project (Frederking & Nirenburg, 1994). Both systems use a number of different MT engines in parallel. Pangloss uses EBMT in conjunction with KBMT (the mainline Pangloss engine) and a transfer-based engine. Of course, there is an element of redundancy in such approaches given that more than one engine may produce the correct translation. Alternatively, one might treat the various outputs as comparative evidence in favour of the best, overall translation. Somers (1999) observes that “what is most interesting is the extent to which the different approaches mutually confirm each other’s proposed translations”.

We do not consider any of these MT systems with multi-engine architectures to be hybrid, in that none of the engines are integrated. At the same time, it is clear that each of the paradigmatic approaches contains favourable elements which, if integrated into a single system, might outperform any of the distinct methods, a view endorsed by many other researchers (cf. Carbonell et al., 1992:235; Lehmann & Ott, 1992:237; Grishman & Kosaka, 1992:263). With respect to EBMT, for instance, there are considerable differences as to the degree that the example-based techniques are used in the overall translation process. Furuse & Iida (1992) have exact matches, as well as generalized templates containing variables. Depending on the level of generalization, some of these templates might be considered notational variants of transfer rules. Other approaches, such as
the ATR system (Sumita et al., 1990), utilise the EBMT module only for specific constructions. A number of other systems (e.g., Sato & Nagao, 1990; Matsumoto et al., 1993) might be described as ‘example-based transfer’ systems (Somers, 1999), in that analysis and generation components are rule-based, but transfer is performed between examples, rather than using rules. Finally, many EBMT systems differentiate between specific and generic examples in order to avoid the problem of example interference. This is very similar to what is done in rule-based systems, where if a specific rule applies, any default rules which could have applied to the same object need to be suppressed.

We intend to show that the integration of DOP’s statistical modelling of language with LFG’s constraint satisfaction into a hybrid language model, LFG-DOP (Bod & Kaplan, 1998), can serve as the basis for a new, robust MT model, LFG-DOT, which promises to improve upon both DOP- and LFG-based MT systems, as well as pure statistical systems and other tree-based systems.

1.3 Current Impediments to Progress in MT

Having described each of the basic approaches to MT, together with their respective advantages and disadvantages, let us now attend to some basic problems which any system will need to confront. There are three main impediments to real progress in MT, namely:

- Ambiguity;
- the Subset Problem;
- the Combination of Exceptions Problem.

We discuss each of these below. We also illustrate each of these problems in chapter 6 with respect to our LFG-DOT models of translation.

1.3.1 Ambiguity

The first of these is, of course, not problematic just in the field of MT, but is a general problem which any NLP system will come up against. Failure to cope with ambiguity—at least to a certain degree—will cause such systems to lose credibility as general solutions. Simply put, ambiguity is a more onerous burden in MT systems than in monolingual parsing systems, for instance, owing to the presence of additional languages which need to be processed. In a transfer-based system, for example, we face all of the monolingual parsing problems in dealing with the source language that any NLP system would face. In addition, there remain the problems of decoding the target language ambiguity as well as coping with the introduction of translational ambiguity.

Traditional rule-based systems have no recourse other than to overgenerate. That is, the number of linguistic objects produced rises with the amount of ambiguity encountered. If this is severe, a linguistic expert will need to sieve through perhaps many hundreds of such objects before selection of the correct object (if there is only one) can be made. Manual inspection is both time-consuming and error-prone, and requires the
availability of highly qualified experts. EBMT, as we mentioned earlier, may introduce spurious ambiguity in adding more examples. If such examples merely reproduce previous examples (either as exact matches or as further instances of generalized templates), the system may be faced with a choice between examples (or fragments) which is irrelevant with respect to the final translation. In any case, the problems of boundary friction and overlapping examples are very real instances of ambiguity. On the other hand, Somers (1999) observes that EBMT systems are able to reduce overgeneration. As they are driven by the data in the aligned corpora, constructions which do not occur can simply be ignored.

Systems which incorporate some level of probabilistic reasoning may obviate the need for manual inspection of the analyses produced for best fit. This means that the selection process proceeds in a completely objective manner: the ‘best’ object is selected purely according to the statistics derived from the corpus used in the application. Of course, the object selected as the best may not be so in practice—it may even be incorrect. Nevertheless, the problems involved in manual inspection are overcome: the mathematical calculations are unlikely to take an overly long period of time to produce, nor are they likely to be incorrect. Finally, the selection process needs no linguistic expert, but proceeds completely automatically.

In addition, an important issue in probabilistic models is pruning. Assuming that the linguistic objects produced can be ranked, then a number of these may be filtered out so that in the extreme case, only one object proceeds to the next level of processing. What is more likely to be the case is that a number of ‘good’ objects are kept (those which exceed some threshold, say) so that the ‘best’ translation is likely to be found in a small number of candidate solutions. Good pruning is informed pruning. The more linguistically sophisticated one’s objects are, the more likely that the search space can be pruned effectively, so that the processing of one’s system benefits from a significant speed-up factor. This issue is discussed further in section 4.6.

Given this, it is our assertion that a desirable quality of MT systems is the ability to output structures with associated probabilities. All else being equal, therefore, systems which embody some kind of probabilistic reasoning are preferable to those that do not.

1.3.2 The Subset Problem

One of the main criticisms of the Rosetta MT system (Rosetta, 1994) is that it advocates tuning of grammars. This requires that grammars for two languages be isomorphic. This is defined as:

1. “For each basic expression in one grammar there is at least one translation-equivalent basic expression in the other grammar;

2. For each rule in one grammar there is at least one translation-equivalent rule in the other grammar.” (op cit., pp.22-23)
Without wanting to give a critique of the Rosetta system here, it is the opinion of this author (at least)\(^7\) that strict adherence to the notion of isomorphy\(^8\) defined above causes monolingual components to be polluted, in that their contents are altered merely because of translational information.

An example is “English wall which is ambiguous because Dutch makes a distinction between *wand* and *muur*” (*op cit.*, p.52). What if other languages were added to Rosetta which made still further distinctions to such words? It would then be bizarre to claim that these words had become still more ambiguous in English. According especially to proponents of the transfer approach, it would be better to include translational information in its proper place, namely in a separate transfer module. In this way systems are designed in a completely modular fashion, causing monolingual source and target components to remain ‘pure’, in that they contain rules applicable only to the particular language of study.

As an example, consider an MT system designed to translate between English, French and German. In building the English-French components, the system designer observes that as a rule, English Adj-N sequences map into N-Adj sequences in French. In order to facilitate the task of transfer, therefore, the decision is taken to produce objects which switch around English adjectives and nouns so that these are in the ‘correct’ order for processing by the French components in the system. However, such objects are not *bona fide* English linguistic objects; rather, they are objects suitable for translation from English to French (and many other languages too, of course). If the same ‘English’ parser is used in the English-German system, then the output of English processing with N-Adj sequences causes problems for the German components, as German adjectives (like English) are prenominal. The German components of this system need, therefore, to unpick the effects of the ‘English’ parser and revert the sequences to what should have been produced in the first place. The bare facts of the matter are that translation of English Adj-N sequences is straightforward for German, and that such sequences need to be inverted (on the whole) when translating into French. If analysis and generation components are restricted to dealing with monolingual information only, then these components in transfer systems remain pure. In the system envisaged here, it is better for the English parser to produce objects which are in every respect linguistically valid for English: in this case, translation of Adj-N sequences into German will be simple, but will require a degree of manipulation when translated into French. However, it is entirely right and proper that the English-French transfer module and/or French generation module deal with this problem, rather than one attempting to ‘solve’ the problem in English analysis, a solution which we have seen introduces problems which are non-existent into the English-German

---

\(^7\)(Syntactic) M-rules are split into (a) translation-relevant; and (b) transformation rules (language-specific). That is, monolingual data is separated from translational data, but not in a separate translation-specific component, as in transfer-based systems. The consequence is that subgrammars, rather than grammars, are now isomorphic, which one would think deviates somewhat from the original claims. Other researchers have thought the same, with one notable observation that this now makes “a complete mockery of the claim that the grammars are isomorphic” (Carroll, 1989). With respect to headswitching, for instance, “(such) types of categorial divergence are a challenge to the isomorphic grammar method” (Rosetta, 1994:230) – indeed, they necessitate significant changes to its architecture. Nevertheless, it proves impossible to keep even subgrammars isomorphic, so that “all subgrammars in one M-grammar must be attuned as far as possible” (*op cit.*, p.252; our emphasis), leading to “partially isomorphic subgrammars” (*op cit.*, p.267), which one might consider to be contradictory.

\(^8\)A further requirement in Rosetta is that grammars be *strictly isomorphic*. This guarantees that a translation is defined for each correct sentence. As can be envisaged, this will not always be the case, so that whilst “the current grammars of the Rosetta system are isomorphic, (and) it is the intention of the grammar writers that they are strictly isomorphic, this has not been proved formally” (Rosetta, 1994:24); nor is this likely, given the later admission that this “cannot be proven” (*op cit.*, p.368).
system.

One would think, then, that there would need to be considerably appealing reasons for actively choosing to tune grammars to one another given the above discussion. Appelo notes that there are two main advantages of tuning:

1. “It is a solution to the ‘subset problem’ of interface structures that all systems using a kind of abstract interface representation have and that guarantees meaning preservation.

2. Only simple transfer modules are needed; the contrastive research is done during the development of the isomorphic grammars.” (Rosetta, 1994:93)

We accept the second assertion without further comment. Let us now describe the subset problem of MT. With respect to an interlingual MT system, for example, it is vital that the structures which are produced by an analysis component be acceptable to the generation component. Appelo (op cit., p.94) illustrates a scenario where this may not be the case with Figure 1.2. Here two source and target grammars map strings into interlingual representations which serve as the interface structure in an MT system. The grammar of SL maps onto a subset \( IL_1 \) of \( IL \), whereas TL maps onto a subset \( IL_2 \) of \( IL \). Consequently, translation is only possible for the strings which are mapped onto the intersection of \( IL_1 \) and \( IL_2 \).

While Figure 1.2 uses an interlingual MT system to illustrate the subset problem, the problem itself is not restricted to interlingua-based systems, but is equally applicable to transfer systems. Appelo observes that:

> “it is not possible to define translation for all ‘possible’ [interface] structures (many of which will not correspond to any sentence at all), but on the other hand it is not possible to characterise what the subset of relevant [interface] structures is and to guarantee their translation” (op cit., pp.94–95).

She concludes that “the only fundamental way to solve this problem appears to be that the grammars of SL and TL are not developed independently, but in close co-operation. In the Rosetta approach, this has been worked out in a specific way: by designing isomorphic grammars.” (op cit., p.95).
Finally, she observes that “most transfer systems do not have a solution to this problem” (ibid.). In particular, she states that the Eurotra MT system cannot properly deal with this problem. This is not altogether true: Eurotra rules consist of translators (which map between levels of structure) and generators (grammars at each level). The task of generators is to consolidate incoming objects. Such objects themselves may be either consolidated (a set of feature bundles with fixed dominance and precedence relations) or unconsolidated (a set of feature bundles which may be subject to further modification, via other translator rules, for instance). Intuitively, therefore, the process of consolidation involves the parsing of incoming objects by the rules of the generator in order to assess the acceptability of objects at all levels. This implies that generators are able to fill in or complete missing information (in terms of other feature bundles) so that dominance and precedence relations may be established.

A (much simplified) example in the Eurotra EF framework would be the insertion of an English VP node in synthesis, via the rule in (15):

(15) \[ S: \{ \text{cat}=s \} [ \ V: \{ \text{cat}=v \}, \]
\[ \quad \text{SUBJ}: \{ \text{cat}=\text{np} \}, \]
\[ \quad \text{OBJ}: \{ \text{cat}=\text{np} \} ]
\[ => \]
\[ S < \text{SUBJ}, V, \text{OBJ} > \]

The angled brackets signify the dominance relation in a descriptor. The descriptor on the right-hand side (RHS) of (15), assuming the presence of typical rules for English surface sentences and VPs, will be consolidated into (16):

(16) \[ \{ \text{cat}=s \} [ \{ \text{cat}=\text{np} \}, \]
\[ \quad \{ \text{cat}=\text{vp} \} [ \{ \text{cat}=v \}, \]
\[ \quad \{ \text{cat}=\text{np} \} ] \]

In Eurotra, VP nodes occurring in surface structure are deleted at higher levels, given the desideratum of flattened, dependency structures at the level of interface structure, where the matrix verb is promoted to the role of governor of the sentence. In generation, the converse is required. Those objects which cannot be consolidated are rejected, and those which are able to be consolidated proceed (via the next set of translator rules) to the next level.

It is clear, therefore, that no matter what system is built, at least one linguistic object created at one level must be acceptable at the next level in the translation process if translation is to be successful. Whether this be via tuning the grammars, as in Rosetta, or via consolidation, as in Eurotra, is perhaps a personal choice. Appelo argues that “Although no quantitative data are available, it is suspected that the trouble of developing interface structures is at least comparable to the extra effort needed to make compositional grammars isomorphic” (op cit., p.96). Unsubstantiated opinion such as this is unhelpful. What is important is that any system have some facility whereby objects are checked for acceptability at the next level, and that failure to overcome such checks results in such objects being deleted.

One might consider that the subset problem affects EBMT systems also. We showed in (12)–(13) how the problem of boundary friction can cause ill-formed target strings. This is exactly the situation with rule-based
systems: strings such as (14) may be built which do not conform to the grammar of the target language. It is not clear that this can be overcome at all in a pure EBMT system, where the examples are stored only as string pairs. Somers (1999) observes that in a hybrid system, however, “one solution ... would be to have a grammar of the target language, which could take the results of the gluing process and somehow smooth them over”. While this may be possible in theory, its implementation is non-trivial. For example, Veale & Way (1997) note that their Gaijin system “offers no comprehensive solution to this problem, but attempts to alleviate it by ensuring that any translation that is recalled from memory ... share(s) the most words (especially marker words) with the previous translation. The intuition here is that in preserving key agreement-carrying words from the original text, the new translation is more likely to slot comfortably into the template”. While perhaps being considered a less than optimal solution, even this attempt is only possible given their use of generalized templates headed by marker words.

### 1.3.3 The Combination of Exceptions Problem

It is regrettable that much of the MT literature is lacking in detail. Many systems are described with respect to simple cases, in addition to some isolated more complex examples. When it comes to handling many more difficult cases, and especially where these co-occur in the same sentence, it is not unreasonable to state that most system designers have nothing to say on this matter; it may even be the case that they do not know how their system would cope with such ‘combinations of exceptions’ at all.

Way et al. (1997) provide a detailed account of how the Eurotra machines (CAT: Arnold et al., 1986; EF: Bech & Nygaard, 1988; and RC: Arnold & Sadler, 1990) cope with ‘hard’ cases, and in particular, combinations of some of these cases. With respect to CAT and EF (the latter the mainstream Eurotra system, a notational variant of CAT), they demonstrate that their translator grammars tend to approximate to sentence dictionaries. That is, where a particular combination of two (or more) linguistic phenomena occur in the same sentence, the individual rules written to cope with each phenomenon in isolation are insufficient to deal with the situation where both co-occur in the same string. In this case, there is no recourse other than to write a new rule capable of handling the situation where both phenomena occur together. The same may be true for another pair of phenomena, so that where all three occur together, yet another rule has to be written.

As an example, Way et al. (1997:343f.) show that CAT (and EF) fails to cope with the translation in (17c):

(17) a. Le gouvernement sait le faire ↔ The government knows how to do it.
   b. On le fait maintenant ↔ It is done now.
   c. On sait le faire ↔ How to do it is known.
   d. On mange bien en France ↔ One eats well in France.

They assume that the translation requires passivisation, given that translating on as one in such examples leads to unnatural English sentences (despite examples such as (17d)), as well as the insertion of how after know in English. Specific rules can be written for the translation of these phenomena in isolation (examples (17a) and (17b)), but where they co-occur in (17c), the individual rules impinge on one another, necessitating
a new rule for this combination of complex transfer cases. This is due to the ‘local tree restriction’ in CAT, whereby if a rule mentions any two nodes in a local tree, it must mention the whole local tree. This can often mean that nodes which play no part in the minimal description of a particular translation problem must still be mentioned. Where such nodes are affected by other rules, these may interfere with other translation rules in which such nodes play only a peripheral part, causing rules to fail.

For example, the CAT rule (18) which translates savoir into know how is forced to mention all daughters of savoir, despite the fact that the subject position has nothing to do with the problem of translating savoir itself:

(18) (?,?,{cat=s}).[\$SAVOIR: (gov,{word=savoir}),
   $1:arg1,$
   (arg2,{cat=s,verbform=infin}).[ $2:*(? )],
   $MODS:*mod ]

=>

(?,?,{cat=s}).[\$SAVOIR,
   $1,$
   (arg2,{cat=s,verbform=infin}).[\$2,
   (mod,{cat=advp}).[{(gov,{word=how})}]]
   $MODS ]

The presence of this node $1$ in the savoir rule prevents the on rule (19)—which states that the voice should be switched and the arg1 not translated—from firing:

(19) (?,?,{cat=s,voice=active}).[ \$G:gov,
   (arg1,{cat=np}).[{(gov,{word=on})}],
   $2:arg2,$
   $MODS:*(? ) ]

=>

(?,?,{cat=s,voice=passive}).[ \$G,
   $2,$
   $MODS ]

The interface representation for On sait le faire is given in (20):

(20) (?,?,{cat=s}).[{(gov,{word=savoir}),
   (arg1,{cat=np}).[{(gov,{word=on})}],
   (arg2,{cat=s,verbform=infin}).[{(gov,{word=faire})},
   (arg2,{cat=np}).[{(gov,{word=le})}]]]

The LHS of (18) will match (20). The node $\$SAVOIR$ will be assigned to the verb governor and $1$ to the NP on. A lexical rule will be found to translate savoir, and le faire is translated quite straightforwardly (the $2$ node on the LHS in (18) is copied over to the RHS with no alteration). The on NP will not be
translated: the LHS of (19) matches structures which are \{cat=s\} with a verb governor, but $1$ in (18) deals with structures which are \{cat=np\} with a noun governor. Given this order of rule interaction, translation of *On sait le faire* fails completely.

The rules (18) and (19) can, however, be applied in the opposite order. (20) matches the LHS of (19). The *savoir* leaf is assigned to $6$, with $2$ assigned to the infinitival clause. The arg1 NP *on* is stripped out. What *CAT* will attempt to produce now is (21):

\[
(21) \quad (?,\{\text{cat}=s, \text{voice}=\text{passive}\}).[(\text{gov},\{\text{word}=\text{know}\}), \\
(\text{arg2},\{\text{cat}=s\}).[(\text{gov},\{\text{word}=\text{do}\}), \\
(\text{arg2},\{\text{cat}=\text{np}\}).[(\text{gov},\{\text{word}=\text{it}\})]]]
\]

The synthesis grammar will then generate (after subsequent application of other rules at other levels) *It is known to do it*, which is not a correct translation of the source sentence. Rules (18) and (19) needlessly encroach on each other’s territory: the rule for *savoir*, (18), tells the machine how to translate the arg1 of *savoir*, by assigning it the identifier $1$ which says “look for a rule that translates this”. This is so even though there is no need for it: the arg1 of *savoir* is not part of the transfer problem surrounding *savoir* and should be left transparent to all other rules. Similarly, the general rule for translating *on*, (19), tells the machine how to translate the *gov* and *arg2* of an *on* sentence, even though they are not part of the complexity involved there and should be accessible to a rule like the *savoir* rule. This encroachment necessitates a new rule (22):

\[
(22) \quad (?,\{\text{cat}=s, \text{voice}=\text{active}\}).[\$\text{SAVOIR}:(\text{gov},\{\text{word}=\text{savoir}\}), \\
(\text{arg1},\{\text{cat}=\text{np}\}).[(\text{gov},\{\text{word}=\text{on}\})], \\
(\text{arg2},\{\text{cat}=s, \text{voice}=\text{infin}\}).[\$2:*(?)], \\
\$\text{MODS}:*\text{mod}]
\]

\[
=> \\
(?,\{\text{cat}=s, \text{voice}=\text{passive}\}).[\$\text{SAVOIR}, \\
(\text{arg2},\{\text{cat}=s, \text{voice}=\text{infin}\}). \\
[\$2, \\
(\text{mod},\{\text{cat}=\text{advp}\}).[(\text{gov},\{\text{word}=\text{how}\})]], \\
\$\text{MODS}]
\]

(22) combines the operations of the *savoir* and *on* rules in a single rule (but of course only applies where the two cases are combined). This effect is pervasive and causes systems like *CAT* to approximate to sentence dictionaries when faced with combinations of difficult translation problems. Way *et al.* (1997) show that more translationally problematic constructions may be included in the same context, as in (23):

\[
(23) \quad \text{On sait permettre que les étudiants le fassent} \implies \text{How to permit the students to do it is known}
\]

In this case, another specific rule will have to be written for the combination of constructions exemplified here.
For completeness’ sake, we point out that a third Eurotra translation machine, Mimo (or RC) (Arnold et al., 1988; Arnold & Sadler, 1990), can handle such combinations of exceptions in a compositional fashion. For instance, a Mimo rule to translate cases with *savoir* would be as in (24):

(24) !savoir.[a!arg2] ⇔ !know.[!mod=how.[a!arg2]]

It can clearly be seen that information about how the subject of *savoir* is to be translated is excluded from this rule. This is quite right in this instance, as a default rule which says ‘translate the source arg1 as the target arg1’ is all that is required here. Such a rule will interact correctly with (24) to produce the correct translation.

The problem of combining exceptional translational phenomena can also affect example-based methods of translation. The analogous position in EBMT might be considered to be the problem of recombining overlapping fragments. Somers et al. (1994) attach ‘hooks’ to each stored fragment which indicate the words and part of speech (POS) tags which can occur before and after the fragment, with an associated weight attached to reflect the frequency of the particular contexts in the corpus. In this way “the most credible combination, i.e. the one with the highest score, should be the best translation” (op cit., original emphasis). Collins (1998) uses the notion of ‘adaptability’ as a measure of the reusability of fragments. Adaptability indicates both the internal structure of a fragment, as well as its external context. Examples must therefore be both a good match for the input string in addition to being a good model (i.e. adaptable) for the output. Finally, statistical modelling of language can be employed to give a confidence factor to strings produced. For instance, with respect to the ill-formed string in (14), p.22, one might incorporate ideas from Grefenstette’s (1999) experiment to seek confirmation for target strings via the web. Somers (1999) performed an informal experiment to search for German strings *Ich sah den*, where *den* indicates accusative case, and *Ich sah der*, where *der* is an error. He found 341 hits for the former match compared to just 17 for the latter. Replacing *Ich* by *ich*, the hits were 467 and 28, respectively.

We therefore consider that any system that is proposed as a general solution to the problems of MT needs to be confronted with ‘hard’ cases of complex transfer, and more importantly combinations of such exceptional phenomena.

1.4 Summary of Thesis

Chapter 2 describes the DOP1 model of language processing, and compares it to other probabilistic alternatives. We note that DOP outperforms other competitive approaches to which it has thus far been compared. We describe more complex DOP models, which are able to deal with unknown words and unknown category words. This latter approach necessitates the manipulation of the observed frequencies of subtrees in order to factor into the probability calculations the fact that there are bound to be a number of unobserved events. We conclude with a number of general observations regarding DOP models.

Chapter 3 describes the LFG approach to MT. It presents the original model of (Kaplan et al., 1989), together with a number of translation examples which are problematic for this model. We also describe other approaches to translation which use LFG, but which ultimately also fail to cope with certain ‘hard’
translation cases, and/or introduce other difficulties.

Chapter 4 describes LFG-DOP, a hybrid architecture for NLP which combines LFG with the probabilistic techniques of DOP. LFG-DOP improves the robustness of LFG, being able to generalize fragments to cope with both ill-formed input, as well as input previously unseen in the training data. We demonstrate why the Discard function of (Bod & Kaplan, 1998) needs to be limited in scope, and offer some suggestions in this regard. We also report on efforts to automatically create the high-quality, large-scale corpora necessary for language processing with LFG-DOP.

Chapter 5 describes previous efforts at using DOP for translation. We focus on two DOT models (Poutsma, 1998; 2000). We show that despite being of interest, there remain a number of phenomena which cannot be dealt with properly by DOT1. DOT2, however, can cope with a range of phenomena that prove problematic for DOT1, but a number of consequences remain.

Chapter 6 presents LFG-DOT, a new approach to translation based on LFG-DOP. We provide four models of translation, together with probability models. We present other possible probability models which have been presented for use with DOP and LFG-DOP models. We demonstrate the impact on corpus size of the Discard function of (Bod & Kaplan, 1998), and describe how this function can be best integrated into our models of translation.

Chapter 7 concludes with a discussion of the contributions of the thesis together with issues for further work.
Chapter 2

The DOP Approach to Language Processing

Data-Oriented Parsing (DOP) language models (e.g. Bod 1992, 1993a, 1993b, 1995, 1998; Sima’an 1995, 1996, 1999; Rajman 1995) assume that past experiences of language are significant in both perception and production. DOP prefers performance models over competence grammars, in that abstract grammar rules are eschewed in favour of models based on large collections of previously occurring fragments of language. New sentences are processed with reference to existing fragments from the treebank, which are combined using probabilistic techniques to determine the most likely analysis for the new fragment.

The general DOP architecture stipulates four parameters on which particular models are instantiated:

1. a formal definition of well-formed representations for sentence analyses;
2. a set of decomposition operations for splitting sentence analyses into a set of fragments;
3. a set of composition operations for recombination of such fragments in order to derive analyses of new strings;
4. a definition of a probability model indicating the likelihood of a sentence analysis based on the probabilities of its constituent parts.

DOP models typically use surface PS-trees as the chosen representation for strings (hence ‘Tree-DOP’, or ‘DOP1’, Bod 1992), but nothing hangs on this choice. We shall later require a more expressive notation, but the use of this notation, while obviously limited, enables the construction of a simple model, applicable to available annotated corpora. The fragments of a Tree-DOP treebank are subtrees, where a subtree $t$ of a tree $T$ consists of more than one node, is connected, and, except for the frontier nodes of $t$, has the same daughter nodes as the corresponding node in $T$.

The parse tree for John eats pizza is (25):
Figure 2.1: The complete Tree-DOP multiset of fragments for the sentence John swims

Given the definition of a well-formed DOP fragment, some invalid fragments derived from (25) include those in (26):

(26) (a) VP (b) VP (c) VP
    S         S         S
    NP        NP        NP
    John      John      John
    V         V         V
    eats      eats      eats
    pizza     pizza     pizza

(26a) is invalid as it consists of just one node, (26b) is ruled out as certain nodes are unconnected, and (26c) is excluded as a possible DOP fragment as the VP has different daughter nodes than in the original tree (25).

The declarative definition of DOP fragments given overleaf can also be described procedurally. Tree-DOP has two decomposition operations to produce subtrees from sentence representations: (i) the Root operation, which takes any node in a tree as the root of a new subtree, deleting all other nodes except this new root and all nodes dominated by it; and (ii) the Frontier operation, which selects a (possibly empty) set of nodes in the newly created subtree, excluding the root, and deletes all subtrees dominated by these selected nodes. Bod & Kaplan (1999:4) point out that this deletion of daughter subtrees “has the effect of preserving the integrity of subcategorization dependencies that are typically encoded as sister relations in phrase-structure
representations”. The complete set of DOP trees derived from the sentence *John swims* are shown in Figure 2.1. Note that all the S-fragments are derived from the full (top left) tree via *Frontier*, and all other fragments are derived via the *Root* operation.

The *composition* operation (◦) enables a tree *T*₂ with a root node *XP* to be substituted at the leftmost non-terminal frontier node of a tree *T*₁ iff that node is also of category *XP* (*T*₁ ◦ *T*₂). The leftmost requirement ensures that the composition of any two subtrees is unique. Given a treebank *B*, a sequence of compositions *t₁ ◦ t₂ ◦ ... ◦ tₙ* where *tᵢ* ∈ *B* which yields a tree *T* with root *S* with no non-terminal leaves, is called a *derivation* of *T*. As an example, assuming the treebank in Figure 2.1, all possible derivations of the sentence *John swims* are shown in Figure 2.2.

Finally, the chosen probability model¹ for Tree-DOP is based quite simply on the relative frequencies of fragments in the treebank, assuming (i) that the trees are stochastically independent; and (ii) that the treebank in question represents the total population of subtrees. Of course, neither assumption is correct, but their adoption allows both the construction of a simple probability model as well as easy subsequent computation. Given that a number of distinct subtree fragments may be derived from a larger tree, it is clear that these subtrees are statistically related, and that their derivation forces these relations to be lost. If all subtrees are taken into account (with no restriction on the depth of tree), the first assumption becomes negligible. The second assumption is problematic when we encounter input which the treebank cannot handle. We shall explore this later when discussing how unknown words might best be dealt with in DOP-based models (see section 2.2).

These elements enable representations of new strings to be constructed from previously occurring fragments in a number of ways. If each derivation *D* has a probability *P(D)*, then the probability of deriving a Tree-DOP representation *R* is the sum of the probabilities of the individual derivations, as in (27):

\[
P(R) = \sum_{D \text{ derives } R} P(D)
\]

Note, therefore, that in Tree-DOP (and, in fact, all DOP-based models) all derivations are mutually exclusive—each needs to be considered in order to derive the probability of each DOP representation. For DOP treebanks of any size, the number of derivations of a particular analysis may be very large. We give some consideration to deriving DOP probability models using a subset of the total number of derivations in sections 4.6 and 6.3.6.

¹Sima’an (1999:9, fn.4) notes that DOP models derive their probabilities using Maximum-Likelihood Estimation (MLE), an instance of inductive learning, where the aim is to generalize a set of training instances into a hypothesis to be tested against new input. The only matter of import in inductive learning is the prior knowledge used for the generalization. He notes more specifically that MLE is an instance of Bayesian learning, where we search for “the most probable hypothesis given the data” (Sima’an 1999:31).

The issue is rather more complex, however. Collins (1999:114) observes specifically with respect to (Bod, 1993) (but generalizable to other DOP models) that “the estimation method is perhaps rather ad hoc – for example not maximizing the likelihood of the training set of example trees ...”. While Collins does not provide a proof, Johnson (1998b) does show definitively that the DOP estimation method is biased and inconsistent, in that the relative frequency estimator of DOP does not maximize the likelihood of the corpus of sentence representations.
Figure 2.2: Derivations for the sentence John swims
The probability of each individual derivation \( t \) is calculated as the product of the probabilities of all the constituent elements \( \langle t_1, t_2 \ldots t_n \rangle \) involved in choosing tree \( t \) from the treebank, as in (28):

\[
P(\langle t_1, t_2 \ldots t_n \rangle) = \prod_{i=1}^{n} \frac{P(t)}{\sum_{t' \in \text{corpus}} P(t')}
\]

An alternative formulation of (28) is (29):

\[
P(\langle t_1, t_2 \ldots t_n \rangle) = \prod_{i=1}^{n} CP(t_i | CS_i)
\]

\( CP(t | CS) \) denotes the probability of selecting a tree \( t \) from a competition set \( CS \) containing \( t \). This competition probability \( CP(t | CS) \) is given by (30):

\[
CP(t | CS) = \frac{P(t)}{\sum_{t' \in CS} P(t')}
\]

\( P(t) \) is the probability of the fragment \( t \). If \( T_{n-1} = t_1 \circ t_2 \circ \ldots \circ t_{n-1} \) is the sub-analysis immediately prior to the \( n^{th} \) step in the whole process, and \( LNC(T_{n-1}) \) denotes the leftmost non-terminal category on the frontier of \( T_{n-1} \), and \( root(t) \) denotes the root category of a fragment \( t \), then the competition set for Tree-DOP at the \( n^{th} \) step is (31):

\[
CS_n = \{ t : root(t) = LNC(T_{n-1}) \}
\]

This shows that the leftmost non-terminal category of the current tree determines the competition sets for Tree-DOP. Furthermore, for any correct derivation it is always the case that \( LNC(T_{n-1}) = root(t_n) \). That is, only a fragment of category \( XP \) can be substituted at the leftmost non-terminal frontier node \( XP \) of a tree. Given this, it is clear that the competition set for any fragment \( t \) depends only on its root node \( root(t) \), which enables us to simplify formula (30) to (32):

\[
CP(t | CS) = CP(t) = \frac{P(t)}{\sum_{t' : root(t') = root(t)} P(t')}
\]

Given that the probabilities are equal to their relative frequencies, an alternative formulation of (32) is (33):

\[
CP(t | CS) = CP(t) = \frac{\#(t)}{\sum_{t' : root(t') = root(t)} \#(t')}
\]

Given these formulae, all derivations for \textit{John swims} are shown in Figure 2.2, with associated probabilities. Ignoring for the time being the topmost derivation (the trivial selection of the full parse tree for the sentence), the probability of the second last derivation is \( \frac{1}{6} \) via formula (28): there is only one subtree \( s[NP \ldots] \).
\( vp[v[\text{swims}]] \) among the 6 subtrees labelled S in the treebank in Figure 2.1, which is then multiplied by 1, there being only one subtree \( np[\text{John}] \) among the subtrees labelled NP. This, however, is not the probability of the sentence \( \text{John swims} \), for we need to sum the probabilities of each derivation in Figure 2.2, as in (27). Given this trivial treebank, this is the only string possible, so \( P(\text{John swims}) = 1 \).

Let us take another simple, yet more interesting example. Given two sentences—\( \text{John swims} \) and \( \text{Peter laughs} \)—and their associated trees, we shall derive the probability of the new string \( \text{Peter swims} \) with respect to this small database of tree fragments. This is the joint probability (i.e. the product of the individual probabilities) of:

1. selecting the subtree \( s[\text{NP } vp[v[\text{swims}]]] \) among the subtrees labelled S;
2. selecting the subtree \( np[\text{Peter}] \) among the subtrees labelled NP.

That is, \( P(t = [\text{NP } vp[v[\text{swims}]]] \mid \text{root}(t) = S).P(t = [\text{np}[\text{Peter}]] \mid \text{root}(t) = \text{NP}) \). These conditional probabilities are computed by dividing the cardinalities of the occurrences of the trees. For instance, \( P(t = \text{np}[\text{Peter}] \mid \text{root}(t) = \text{NP}) = \frac{\#(\text{np}[\text{Peter}] \mid \text{root}(t) = \text{NP})}{\#(t \mid \text{root}(t) = \text{NP})} \).

The complete collection of fragments derivable from the sentences \( \text{John swims} \) and \( \text{Peter laughs} \) are shown in Figure 2.3. Given this small corpus, \( P(t = [\text{NP } vp[v[\text{swims}]]] \mid \text{root}(t) = S) = \frac{1}{17} \). That is, there are 12 trees possible with \( \text{root} = S \), only one of which—the top right subtree—matches this structure.

The copies of trees are produced in different ways: for instance, the tree pattern \( s(np(_), vp(_)) \) is a subtree of the full parse tree of both example sentences. Of course, given an analysis such as (34), we see that this pattern occurs twice, so tree (34) will contribute both examples as subtrees relevant for the processing of further appropriate examples:

\[
(34) \quad S \\
\quad NP \quad VP \\
\quad John \quad thought \quad COMP \quad S \\
\quad that \quad NP \quad VP \\
\quad Mary \quad V \quad won
\]

Importantly then, we see that a DOP treebank is a bag, rather than a set. Each tree which can play a part in combining together with other trees to form a representation for a sentence is used to contribute to the overall probability of that representation given the corpus. Note also that (34) shows DOP to be a productive, recursive model of language.
Figure 2.3: The complete set of fragments derivable from the sentences John swims and Peter laughs
It is not difficult to see that \( P(t = [\text{np}[\text{Peter}]] \mid \text{root}(t) = \text{NP}) = \frac{1}{2} \), as there are only 2 trees possible with \( \text{root} = \text{NP} \) in Figure 2.3, one for \( \text{Peter} \) and one for \( \text{John} \). As a result, therefore, \( P(\text{Peter swims}) = \frac{1}{12} \cdot \frac{1}{2} = \frac{1}{24} \), assuming this derivation. This probability is small, and does not reflect the probability of the string \( \text{Peter swims} \) given this treebank, as the probability of a parse-tree is computed by considering all its derivations, as in (27), which can also be written as (35):

\[
(35) \sum_i j \frac{\#(t_{ij})}{\#(t \mid \text{root}(t) = \text{root}(t_{ij}))}
\]

There are 7 such derivations possible, as shown in Figure 2.4. This gives \( P(\text{Peter swims}) = \frac{40}{192} = \frac{5}{24} \), reflecting the fact that of the 4 possible sentences derivable using this toy corpus, the original sentences \( \text{John swims} \) and \( \text{Peter laughs} \) are more probable (\( \frac{7}{24} \) each) than the new derivable strings \( \text{Peter swims} \) and \( \text{John laughs} \). The reason for this is that full trees exist for the original strings (with probabilities \( \frac{1}{12} \) each), DOP finds these as well as the remaining trees labelled S used to derive the other two sentences, hence the difference of \( \frac{2}{24} \), or \( \frac{1}{12} \). Each of these sentences is unambiguous with respect to the corpus, so the conditional probabilities of their parse trees, by a vacuous application of formula (36), is 1:

\[
(36) P(T \mid T \text{ yields } S) = \frac{P(T)}{\sum_{T'} \text{ yields } S P(T')}
\]

That is, the probability of a parse tree \( T \) given that it yields a sentence \( S \) is the probability of \( T \) divided by the sum of the probabilities of all parses that yield \( S \). Of course, there will be other sentences which are ambiguous in other treebanks, where many different representations will be derivable, each with different probabilities. For example, assuming the two trees in (37), two different parse trees can be derived for the sentence \( \text{John likes flying planes} \):

\[
(37)
\]

These two analyses are given in (38):
Figure 2.4: Possible derivations of *Peter swims* given the Treebank in Figure 2.3
Assuming a collection of fragments built solely from the trees in (37), Tree-DOP will assign a higher probability to the tree on the right as it occurs as a full parse tree in (37), as well as being capable of assembly via smaller fragments, as in (39), for example:

The leftmost tree in (38), however, can only be constructed via smaller fragments, such as in (40), for instance:

This means that there will be more derivations for the rightmost tree in (38), including its instance as a full parse tree, which results in this tree having a higher probability than the other.

Despite the small treebanks used, the illustrations are provided here to give the reader a flavour of the simplicity of the DOP approach given a probability model based on relative frequency, as well as the preference in DOP (in terms of higher probability) for larger trees (as shown in (37-40)). We shall take advantage of this later when it comes to translation, as the same effects are seen there too. Nevertheless, we also report in sections 2.2.3 and 6.3.5 on some unfortunate repercussions due to the adoption of relative frequency as the probability model for DOP.
2.1 A Comparison of DOP with other Probabilistic Techniques

Before we examine how DOP would work as a model for MT (both on its own as well as in harness with LFG), we should satisfy ourselves that it brings more to bear to the problem than other statistical approaches.

2.1.1 Why Statistics?

Why are probabilities used in the first place? Feynman et al. (1963) make the following observation:

“We make guesses when we wish to make a judgement but have incomplete information or uncertain knowledge .... the theory of probability is a system for making better guesses. The language of probability allows us to speak quantitatively about some situation which may be highly variable, but which does have some consistent average behaviour” (our emphasis).

Given this, Ney (1997:1) points out with respect to NLP (at least):

“In our applications, the statistical models are simplifications of complex dependencies in the real world of speech and language ... Often this goes hand in hand with the desire to have a parsimonious description of the relevant dependencies in speech and language data”.

What this means in practice is that depending on the approach chosen, some statistical approaches to NLP may be founded on a number of wrong premises:

i. that language is a stationary, ergodic process

There really is no way around this, of course. For us to produce statistical models, we must assume that the data to be sampled constitutes a closed set, i.e. that it is complete and correct. There are, nevertheless, many techniques (e.g. Good-Turing estimation, cf. section 2.2.2) which permit one’s probabilities to be smoothed (i.e. adjusted) on the assumption that unseen data is necessarily excluded from one’s sample. We discuss this further below. Notwithstanding this, adopting the assumption that language is a stationary, ergodic process enables cross entropy to be used to critically evaluate the quality of statistical language models (e.g. Chen & Goodman 1996). Given the large data sets necessary to produce accurate probabilistic accounts of language phenomena, in practice smaller random samples are abstracted from the data to produce increasingly refined statistical models, thereby avoiding the computational overload involved in using all available data. We report further in section 4.6 on attempts at improving the efficiency of DOP models.

ii. that the probability of the next word depends only on the previous k words

The adoption of this principle allows n-gram models and Hidden Markov Models (HMMs) to be used, especially in POS-tagging (DeRose 1988; Church 1988; Cutting et al., 1992). Of course, studies have been performed showing that related words (such as collocations, or predicate-argument relations) may occur large distances apart. Here the problem is what size window to use in clustering experiments (e.g. Grefenstette 1993, uses a 10-word window; Gale et al. 1992, 1000 words; Brown et al. 1992b, 1001 words), but in
order to reduce the mathematical complexity of the calculations involved, most studies rely on bigrams and trigrams. Given that such techniques explicitly exclude anything but local context from the decision process, a preference is often to use Probabilistic Context-Free Grammars (PCFGs) or Stochastic CFGs (SCFGs) “to escape the linear tyranny of n-gram models and HMM tagging models ... language has a complex recursive structure and such tree-based models – unlike Markov models – allow us to capture this” (Manning & Schütze, 1999:381).

In practice, however, PCFGs provide us with worse language models than n-gram models do, as they are unable to factor in lexical co-occurrence. Furthermore, PCFGs are necessarily based on a linguistic theory, and are therefore restricted by what is contained in the rules of whatever theory is used. DOP has no such locality restriction, in principle at any rate (although we note that the number of resultant fragments is very large), enabling collocations to be captured which naturally occur outside of PS-rules. As an example, consider the sentence the emaciated man starved, where one of the fragments obtained would be (41):

(41) S
   NP     VP
     Det Adj N V
         emaciated starved

Many DOP fragments correspond exactly to PS-rules, but this need not be the case, as we see in (41). It is often stated (e.g. Collins, 1999:75; Manning & Schütze, 1999:387) that PCFGs fail to capture lexical dependencies, but this needs to be tempered somewhat: while a CF-rule such as \( S \rightarrow \text{Det emaciated N} \) could be written, in doing so one loses the internal, hierarchical structure of trees like (41), so maintaining lexical relations in PCFGs is avoided in practice.

Sima’an provides a further criticism of PCFGs:

“Extending [linguistic-based grammars] with probabilities does not make them suitable for modeling linguistic input-output behavior of humans ... Moreover, there is a related fundamental question with regard to whether probabilities should be attached to linguistic competence grammar rules in the first place. Generally speaking, probabilities are more significant when they are attached to dependencies and relations that are more significant for the task they are employed for. In language modeling, often relations between words, between phrasal-categories and, for some sentences, even between whole constituents are most significant in determining the correct analyses of sentences. Therefore, attaching probabilities to the rules of a linguistic competence grammar is equivalent to attaching them only to a small portion of the linguistic relations that are significant to syntactic and semantic analysis” (Sima’an, 1999:6-7, original emphasis).

Nevertheless, one advantage of PCFGs over HMMs is their assignment of higher probabilities to longer strings. For example, a PCFG would assign a higher probability to the string John baked a cake than it would the string John baked a, given the preference for a rule such as \( \text{VP} \rightarrow \text{V \ DET \ N} \) over one like \( \text{VP} \rightarrow \)
V DET, whereas we can expect the opposite result to be achieved with an HMM. Given the implausibility that \( P(\text{cake} \mid \text{John baked a}) \) would have a probability of 1 in a corpus, an HMM would assign a greater probability to \( P(\text{a} \mid \text{John baked}) \) than to \( P(\text{cake} \mid \text{John baked a}) \). In any case, the probability of longer strings such as these can only ever be at most equal to the shorter strings on which they are based, and even this is extremely unlikely.

iii. that \textbf{words in context are independent of one another}

“This simplifying assumption makes it possible to adopt an elegant model that is quite effective despite its shortcomings” (Manning & Schütze 1999:237), namely Bayesian techniques, especially widely used in word sense disambiguation tasks (whether this be \textit{supervised}, e.g. Brown \textit{et al.} 1991; Gale \textit{et al.} 1992; \textit{dictionary-based}, e.g. Lesk 1986; Yarowsky 1992; Dagan & Itai 1994; or \textit{unsupervised}, e.g. Schütze 1992; Schütze & Pedersen 1995).

Despite these criticisms, such statistical models do a good job compared to rule-based models. Nevertheless, a number of basic problems remain to be resolved, regardless of the specific approach adopted, namely:

1. \textit{Search}: Given the large number of words involved, it is often impossible to perform a full search, but sub-optimal searches “find the global optimum in virtually all cases” (Ney 1997:6);

2. \textit{Modelling}: Given the large number of parameters involved in a model, these must be reduced “by taking suitable data dependencies into account”, i.e. capture the most relevant, and exclude those having least effect on one’s statistics. For example, maximum entropy might be used to initially select features, followed by the cross-validation of each feature against samples of data withheld from the initial dataset. “If the feature does not lead to an increase in likelihood of the withheld sample of data, the feature is discarded” (Berger \textit{et al.}, 1996:51; cf. also Abney’s (1997) investigation into probability models based on random fields in section 6.3.5).

3. \textit{Training}: Those techniques based on parameter estimation—a complex mathematical optimization problem—can also prove computationally inefficient.

4. \textit{Sparse data}: “Maximum Likelihood Estimation is in general unsuitable for statistical inference in NLP. The problem is the sparseness of our data ... One might hope that by collecting much more data that the problem of data sparseness would simply go away ... In practice, it is never a general solution to the problem” (Manning & Schütze 1999:198).

Of course, there are a number of well-known solutions to these problems, namely:

1. “The secret to avoiding this complexity (i.e. that direct evaluation of decoding is intractable) is the general technique of \textit{dynamic programming} (or \textit{memoization}) where we remember partial results rather than recomputing them” (Manning & Schütze 1999:326), i.e. the probability of longer subpaths can be computed in terms of one shorter subpaths, using forward or backward procedures, which are much cheaper algorithms.

2. The Viterbi algorithm finds the best path through the search space, i.e. it computes the state sequence in an HMM which best explains the training data (that is, makes it most likely).
3. The Forward-Backward algorithm (or Baum-Welch re-estimation) guarantees to find a better (though not necessarily the best, given that it might get stuck at a local, rather than global, maximum) estimation of the parameters of the model, using the technique of Expectation Maximization (EM algorithm).

4. There exist a number of ways of normalizing (or smoothing) the probabilities, e.g. Maximum Likelihood Estimation, Cross-Validation, and Good-Turing Estimation, all of which use nothing more than the raw frequency of n-grams to obtain the best estimate of their probability in unseen text; or Linear Interpolation and Backoff Models, which combine estimations to circumvent the problem of sparse data to a large extent.

Good-Turing Estimation is described in detail in section 2.2.2, this being an integral part of the DOP3 model (Bod 1995, 1998), which attempts to deal with the problem of unknown words.

2.1.2 Why DOP?

Taking on board these criticisms of statistical processes, with respect to DOP (and LFG-DOP in particular, as we shall see later), there is no doubt that the search problem is an onerous one. Despite this, and especially given most previous attempts at MT, the onus remains on us to find a better methodology for the solution of this problem; if the theory is wrong to start with, we can anticipate that any resulting attempted solution will fall short of the mark. In any case, certain of these well-known techniques cannot be used in DOP. For instance, we would like to be able to use the most probable derivation rather than the most probable parse to be able to use Viterbi optimization, but Bod & Scha (1997:158-162) observe a general degradation in performance in this case:

- Parse Accuracy: decreased from 64% to 46%;
- Sentence Accuracy: decreased from 75% to 68%;
- Bracketing Accuracy: decreased from 94.8% to 92%.

The Viterbi algorithm is used a lot in HMM-based tasks to find the most likely parse (e.g. DeRose 1988; Cutting et al. 1992; Merialdo 1994), or word-sense (e.g. Gale et al. 1992; Yarowsky 1992) in cubic time. When a Stochastic Context-Free Grammar (SCFG) is used, the most probable derivation is the same as most probable parse (Bod 1995:61, who also notes that where the depth of subtrees is restricted to 1, then the DOP model is equivalent to a SCFG where each parse is generated by exactly one derivation), but if a Stochastic Tree Substitution Grammar (STSG) is used, the probability of the parse is the sum of the probabilities of all derivations.² Again we note that SCFGs are weaker than STSGs, as the former run into trouble when faced with dependencies beyond the scope of single rewrite rules, as in (41).

The basic idea of the Viterbi algorithm is to eliminate low probability sub-derivations in a bottom-up manner. If two different sub-derivations produce parses of a substring with the same root, the one with the lower probability can be eliminated. Bod (1998:44) illustrates this with the two simple trees in (42):

²cf. Bod (1998:26-27) for a proof that DOP is indeed equivalent to an STSG.
If \( P(D1) > P(D2) \), we can eliminate \( D2 \) if we are trying to find the most probable *derivation* of *abcd*, but if we want the most probable *parse* of *abcd*, then \( D2 \) cannot be deleted, as there may be other elementary trees which need \( D2 \) but not \( D1 \), as in (43):

What this amounts to is that a best-first search is not adequate for finding the most probable parse in STSGs, so there are no deterministic polynomial time algorithms available for solving this problem (i.e. it is NP-hard). However, several optimizations of DOP have been proposed which may carry over to the field of MT. For instance, by restricting the subtrees to those with maximally two non-terminals, Sima’an (1995; 1999; cf. also sections 4.6 and 6.3.6) produces results for the *Air Travel Information System (ATIS) Corpus* (Hemphill et al., 1990; Marcus et al., 1993) where the most likely parse and the most likely derivation are very close to one another. Furthermore, by iteratively generating a large number of random derivations via the Monte-Carlo method (assuming these are based on the actual probabilities of the sub-derivations), Bod (1995:51) shows that the most probable parse can be estimated as the parse that results most often from these random derivations (by the law of large numbers). Nevertheless, Sima’an (1999:11, fn. 6) notes that “although Bod (Bod, 1995) claims that his algorithm is non-deterministic polynomial-time, (Goodman, 1998) shows that Monte-Carlo parsing is exponential-time”. Bod (personal communication) notes that he has never claimed that that Monte-Carlo parsing was ‘non-deterministic polynomial-time’. For instance, “we will leave it as an open question whether the most probable parse can be deterministically derived in polynomial time” (Bod, 1995:50). Bod (*op cit.*) goes on to show that Monte-Carlo *estimation* of the most probable parse is cubic in sentence length. Sima’an (1999:105) observes that this too has been disputed:

“Goodman (1998) argues that the error-rate in the time-complexity of the Monte-Carlo algorithm is actually not a constant but rather a function of sentence length: this implies that Monte-Carlo parsing has in fact exponential time-complexity”.

Sima’an observes (1999:105, fn. 2) that “although the argument in (Goodman, 1998) does not constitute a full proof, it is sound and convincing”. Bod (personal communication) counters that Goodman’s (1998) argument is convincing in terms of the worst-case time complexity, but given Zipf’s law, it is not. We shall revisit Sima’an’s improvements in the efficiency of DOP models in greater detail in section 4.6.

De Pauw (2000) reinterprets DOP as a pattern-matching model which tries to maximize the size of the fragments participating in the analysis of a sentence, rather than the probability of the parse. This eliminates
<table>
<thead>
<tr>
<th>Disambiguator</th>
<th>Parse Accuracy (/562)</th>
<th>%</th>
<th>$F$</th>
<th>Parse Accuracy on parsable sentences (/456)</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCFG</td>
<td>373</td>
<td>66.4</td>
<td>83.0</td>
<td>373</td>
<td>81.8</td>
</tr>
<tr>
<td>PMPG</td>
<td>327</td>
<td>58.2</td>
<td>75.1</td>
<td>327</td>
<td>71.7</td>
</tr>
<tr>
<td>PCFG + PMPG</td>
<td>402</td>
<td>71.5</td>
<td>85.2</td>
<td>402</td>
<td>88.2</td>
</tr>
</tbody>
</table>

Table 2.1: Results of De Pauw’s (2000) experiments using Pattern-Matching

the notion of multiple derivations, which allows for efficient Viterbi-style optimizations not possible with DOP. He notes that while DOP achieves a very high parse accuracy, Bod (1995) reports that the average parsing time was 3.5 hours per sentence.\(^3\) The major reason for both these facts is the implicit, necessary redundancy in DOP models: multiple copies of tree fragments contribute to the probability model in terms of their relative frequency, so multiple copies of the same parse-tree are built. CFGs, on the other hand, can only build a particular structure in one way. De Pauw proposes to mimic what may be the way humans process language by retrieving the most similar structure to the proposed analysis from memory. Similarity is computed according to the number of patterns needed to construct a tree (to be minimized) and the size of the patterns used to construct the tree (to be maximized).

De Pauw compares three approaches of disambiguation: PCFG, Pattern-Matching Probabilistic Grammar (PMPG, which he terms ‘the DOP approximation’), and a combined, integrated PCFG + PMPG system, in terms of parse accuracy as well as the usual figures for precision $P$ and recall $R$ interpreted by the $F$ measure, defined as in (44):

$$
F = \frac{1}{\frac{1}{P} + \frac{1}{R}}
$$

De Pauw does not state what value of $\alpha$ he uses, but if it is the usual 0.5 then (44) simplifies to $\frac{2PR}{P+R}$. De Pauw is interested in sequences of POS-tags, rather than word strings, and limits the experiment to 562 sentences of the ATIS corpus with a maximum sentence length of 15 words. 106 of the 562 sentences (19%) could not be parsed by his bottom-up chartparser owing to the sparseness of the grammar, especially for NPs. Hence the maximum parse accuracy for any of the disambiguation models is 81%. The results are presented in Table 2.1. The PCFG achieves 66.4% parse accuracy on the whole test set, and 81.8% on those sentences which are parsable. However, analysis of the errors shows that PCFGs tend to prefer flatter structures over embedded ones. The correct analysis for *I want a flight from Brussels to Toronto* is given in (45), along with the parse suggested by the PCFG:

\(^3\)Bod (personal communication) says that this figure was obtained on a Sparc 2 for the case where all subtrees were used. With a current Sparc 60, for instance, the same experiment takes only a few seconds per sentence.
This is to be expected as flatter structures obviously require fewer rules than more hierarchical ones. De Pauw notes that “it is an unfortunate property of PCFGs that the number of nodes in the parse tree is inversely proportionate to its probability”, and suggests that the effect of this may be overcome by normalizing the probability of a parse tree relative to the number of nodes in its tree. However, he also notes that a more linguistically sound alternative is available, namely using larger syntactic context within the parse trees. Bod (2000b) also shows that a non-probabilistic version of DOP outperforms probabilistic DOP on the ATIS and OVIS corpora, while obtaining competitive results on the Wall Street Journal (WSJ) corpus. He claims that this overturns the conventional wisdom that ‘the bias of stochastic grammars in favour of shorter sentence derivations is harmful and should be redressed’, at least for STSGs.

De Pauw’s PMPG model is inspired by (Goodman, 1996) who “unsuccessfully uses a system of indexed parse trees to transform DOP into an equivalent PCFG”. Despite this, De Pauw adapts Goodman’s system of indexing trees to encode contextual information in parse trees. Given an indexed training set, De Pauw matches indexes on a parse tree from the test set in a bottom-up manner. For instance, in (46), where boxed nodes indicate nodes retrieved from memory, the fully specified fragment \( N_P(N_P,P_P(\text{in},N_P)) \) has been retrieved from memory, but no fragment \( V_P(vbp,N_P) \) was found containing exactly this retrieved NP:
The disambiguation phase of PMPG involves the pruning of all nodes retrieved from memory, which results in the pruning of (46) to (47):

(46) !\[S \rightarrow \text{NP-SBJ} \rightarrow \text{VP} \rightarrow \text{NP}

(47) !\[S \rightarrow \text{NP-SBJ} \rightarrow \text{VP} \rightarrow \text{NP}

The probability of the pruned tree (47) is computed in the usual PCFG manner, as in (48):

(48) !\[P(\text{parse}) = P(S \rightarrow \text{NP-SBJ VP}).P(\text{VP} \rightarrow \text{vbp NP})

The results for PMPG disambiguation in Table 2.1 show that this ‘context-sensitive’ parsing model performs significantly worse than the PCFG model, indicating that PMPG cannot accurately reflect the built-in context sensitivity of DOP. In analysing the types of error made by PMPG, De Pauw observes that this model overestimates sub-structure size as a feature for disambiguation.

The combined PCFG+PMPG model achieves a parse accuracy of 81.8%, and De Pauw notes that the correct parse is included in the 10 most probable parses 99% of the time. De Pauw states that it might be possible to generate only the best 10 parses using a best-first strategy and use these for disambiguation. The combined PCFG+PMPG model outperforms both the PCFG and PMPG models, but no analysis is given of the remaining errors made by the system.

Despite the optimization problem, which all statistical techniques run up against, one criticism of other approaches which cannot be levelled at DOP is that they cannot handle non-local phenomena. For instance, Carroll & Weir (1997) compare Tree-DOP with SCFG (Booth & Thompson, 1973), Stochastic Lexicalized Tree Adjoining Grammar (SLTAG, Schabes 1992)/Probabilistic Tree Adjoining Grammar (PTAG, Resnik 1992), and Link Grammar (Lafferty et al., 1992). Their results show that whilst Tree-DOP is able to stochastically differentiate all the structural units that the other frameworks can, Tree-DOP, on the other hand, is the only one able to statistically relate arbitrarily far apart constituents, illustrating this claim with respect to example (41), p.45.

The fact that they cannot handle non-local phenomena is a general criticism of length-based approaches, whether we are using an n-gram approach (e.g. Brown et al., 1990), or a window-based approach (e.g.
Grefenstette 1993; Brown et al., 1992b; Gale et al., 1992). In attempting to deal with such phenomena, these techniques resort to ad hoc procedures in so doing, e.g. Candide (Brown et al., 1992a) captures lexical dependencies using the notion of 'potential informant'. For a verb, this might be '1st NP to the right'. Whether we are dealing with collocations, or long-distance dependencies, or trying to establish the best chunk for mapping in translation, tree-based approaches such as DOP handle such phenomena within structures. Such dependencies have nothing to do with length, or distance, which are arbitrary, wrong notions, nor complex notions like mutual information.

The major benefit of the DOP approach over all other statistically based techniques is that no intensive training cycle is required, and despite this DOP outperforms all other models it has so far been compared with (Bod 1996; Charniak 1996; Sima’an 1996). In his work, Charniak states that his system “outperforms all other non-word-based statistical parsers/grammars on [the Wall Street Journal] corpus” (Charniak 1996:1031), despite using only relative frequency rather than sophisticated re-estimation techniques. This calls into question the merits of sophisticated training or learning algorithms, such as Inside-Outside Re-estimation (Pereira & Schabes, 1992, who perform bracketing experiments on the ATIS corpus), or Transformation-Based Error-Driven Learning (TBEDL), which has been used for POS-Tagging (Brill 1992, 1994), building POS-trees (Brill 1993), Text Chunking (Ramshaw & Marcus, 1995), and resolving PP-attachment (Brill & Resnik, 1994). Despite claims that TBEDL produces results which are comparable to HMMs, yet which provide more compact and perspicuous models, as well as the fact that they can be implemented using finite-state techniques (Roche & Schabes, 1995), it must nonetheless loom out in any comparison with DOP, where elementary trees are read directly from a treebank of parsed strings, and new trees are assigned probabilities purely on the basis of the raw frequency of subtrees in that treebank. One might be able to make a case for TBEDL if it outperformed DOP, but it does not. Similarly, Goodman (1998:179) shows that Tree-DOP “achieves an average zero-crossing brackets accuracy” to a statistically significant level (an error reduction of 1/3) over the method of (Pereira & Schabes, 1992).

Perhaps the nearest approximations to the DOP approach are PTAG (Resnik 1992) and SLTAG (Schabes 1992). It is an interesting question as to whether there remain any constructs that DOP cannot handle that these similar approaches can, i.e. which require adjunction, rather than merely tree substitution.4

This adjunction method of composition is the only difference between DOP and the TAG approaches, which makes SLTAG (and PTAG) mildly context-sensitive. Given also that SLTAG is derived from an underlying grammar, certain word dependencies (such as (41), p.45) cannot be captured by SLTAG, as only one lexical ‘anchor’ is permitted in each elementary tree, so statistical dependencies between words cannot be expressed. DOP, on the other hand, can handle such collocational information quite naturally. Finally, only the probability of a derivation can be calculated given current probabilistic implementations of TAG (Resnik

---

4One example may be the French negation elements _ne_ ... _pas_. Nevertheless, DOP will produce trees which contain both _ne_ and _pas_, so this collocation would seem to be treatable in DOP-based approaches. Furthermore, a purported advantage of TAG over TIG is that TAG can have ‘wrapping adjunction’. Schabes & Waters (1993) give _deduce from_ as an example, as in:

(49)    John deduces that Mary invited Bob from smelling smoke.

However, DOP again would permit fragments containing nothing other than _deduce_ and _from_ (among other relevant fragments), allowing the link between these lexical items to be maintained. In any case, it is disputable whether wrapping adjunction is needed to model most languages. Dutch cross-serial dependencies would be an exception, and DOP1 cannot handle these examples.
1992; Schabes 1992), and we need to know also what the probability of a tree, and perhaps its meaning, is. Currently, therefore, SLTAG/PTAG is restricted by the linguistic dependencies of the underlying grammar (a competence model), i.e. the statistical dependencies of SLTAG/PTAG are the linguistic dependencies of the grammar, whereas DOP clearly favours a performance model.

Given the problem that only the probability of a derivation is accounted for and not the probability of an analysis with TAG models, this, together with allowing for multiple anchors, provides the motivation for ‘TIGDOP’ (Hoogweg, 2000), a version of DOP which allows both insertion, as in Tree Insertion Grammars (TIG: Schabes and Waters, 1995) and substitution, as in DOP, as composition operations. Hoogweg notes that a problem with Tree-DOP is that while DOP trees may encode statistical dependencies, their subsequent fragmentation may cause these dependencies to be lost. For instance, in DOP it is impossible to combine the two trees in (50) to parse the grey parrot whistled, so the link between parrot and whistled is lost.\footnote{N↓ in (50) signifies that the N is a possible substitution site.}

![Diagram](image)

Furthermore, it would be impossible for the parrot whistled to be derived from trees sourced from the grey parrot whistled, as DOP cannot remove the modifier plus its associated category from such fragments. Given these problems, Hoogweg suggests that they may be overcome by extending DOP with the insertion operation.

TIG is exactly like TAG except that TIG does not allow elementary wrapping trees or elementary empty auxiliary trees. “This ensures that every elementary auxiliary tree will be uniquely either a left auxiliary tree or a right auxiliary tree. (Wrapping auxiliary trees are neither. Empty auxiliary trees are both and cause infinite ambiguity,)” (Schabes & Waters, 1995:481). Furthermore, adjunction is prohibited in TIG at certain nodes which ensures that TIGs only derive CF-languages, so that cubic algorithms used for parsing CFGs can be adapted for use with TIGs. As TIGDOP is a version of DOP, it is defined using the usual four system parameters. The utterance analyses are DOP trees. The definition of subtrees is similar to the DOP subtree definition except for an additional condition which states how auxiliary structures may be removed from trees. A TIGDOP elementary subtree of a tree $T$ is a subgraph $t$ of $T$ such that:

- $t$ consists of more than one node, and
- $t$ is connected, and
- for every node $\mu$ in $t$ it holds that
  - $\mu$ has no daughter nodes of the corresponding node in $T$, or
  - $\mu$ has all the daughter nodes of the corresponding node in $T$, or
– \( \mu \) has all the daughter nodes of a left- or rightmost descendant of \( \mu \) in \( T \) with the same label as \( \mu \).

The first two conditions are the same as in Tree-DOP (cf. (26), p.35, and resultant discussion). In order to illustrate the effect of the third, new condition, let us assume the tree in (51):

\[
\text{S} \\
  \text{NP} \quad \text{VP} \\
  \text{John} \quad V \quad N \quad PP \\
  \text{ate} \quad \text{dinner} \quad P \quad \text{NP} \\
  \text{with} \quad \text{Mary}
\]

(51)

Now the tree in (52) is a valid elementary subtree of the tree in (51), as the PP auxiliary structure with Mary can be removed from (51):

\[
\text{S} \\
  \text{NP} \quad \text{VP} \\
  \text{John} \quad V \quad N \\
  \text{ate} \quad \text{dinner}
\]

(52)

Under Tree-DOP, (52) could never be derived from (51).

As stated, insertion and substitution are the composition operations of TIGDOP. The probability calculation is somewhat more complex than in Tree-DOP, as the probabilities of selecting left and right auxiliary trees from a corpus have to be factored in, as well as estimating the probabilities that insertion and adjunction take place at appropriate nodes (cf. Hoogweg, 2000:29–31).

Hoogweg shows that Tree-DOP sampling does not ensure that a unique random derivation is produced in some cases. We describe this in section 5.2.3 with respect to the DOT2 translation system. Like De Pauw (2000), Hoogweg jettisons terminals in order to try to overcome problems of data sparseness. To establish a baseline, he constructs a DOP1 model which shows best accuracy results at depth 5 (51.52\% exact match) on a test set of 100 ATIS trees. Hoogweg also discusses issues pertaining to lexicalization of fragments, showing that TIGDOP models suffer less from missing coverage than DOP models, although in the context of his word-less experiments this is a little odd.

For TIGDOP1, there is a considerable increase in grammar size compared with DOP1: at maximum depth 10, there are 26852 DOP1 trees, but 41793 TIGDOP1 trees (an increase of 56\%). Furthermore, Hoogweg’s results show a decrease in accuracy for all fragment depths. Part of the problem may be some of the strange trees produced: from the sentence \( S(NP(John),VP(V(boarded),NP(DET(the)),N(plane)),PP(p(to)),N(Paris))) \),
TIGDOP1 correctly produces the fragments $s(NP(John),VP(v boarded),NP(DET(the),N(plane)))$ and $NP(NP*,PP(p(to),N(Paris)))$. However, it also allows $s(NP(John),VP(v boarded),NP(Paris))$ by extraction of the structure of the phrase modifying Paris, namely the plane to. This leads to a new model TIGDOP2, which permits only the extraction of auxiliary trees with spine of length one. The results show that TIGDOP1 scores better on exact match, but TIGDOP2 does better on recall. The major benefit of the TIGDOP2 model is that its best results are achieved at maximum depth 5, whereas TIGDOP1 obtains some of its better results at levels up to depth 10. Hoogweg concludes that TIGDOP2 enables the restriction of trees, but compared to DOP1 it still produces 30% more tree fragments at depth 5 with worse results.

One final adjustment to the probability model to prevent certain structures being derived in different ways (via insertion and substitution) results in TIGDOP3. This leads to large increases in accuracy over TIGDOP2, especially for exact match, in which category it also outperforms DOP1 (60.61% vs. 56.57%). However, retrieval of fragments in DOP1 in terms of precision and recall scores much more highly than any TIGDOP models. As such, Hoogweg concludes that TIGDOP3’s results “are less robust than those of the DOP1 model”. Nevertheless, his results confirm a preference for DOP over LTAG models which cannot model statistical dependencies between words.

Bod (1998:37-38) compares Tree-DOP to head-lexicalized stochastic grammars (HLSG; Collins 1996, 1997; Eisner 1996, 1997; Charniak 1997). In such formalisms, each non-terminal in a tree is associated probabilistically with the head word of the constituent. Such grammars are richer than SCFGs. Nevertheless, they do not cover dependencies involving non-heads. Bod (1998) illustrates the problem with sentence (53):

(53) Show me the nearest airport to Dallas

Here the PP can be attached to either show or to airport, so HLSG would compare the probabilities of these two attachments. However, in this example, the appropriate attachment depends on nearest, which is not a head in either case (cf. Bod (2000b) for an experimental comparison of DOP with HLSG on such phenomena).

In the WSJ corpus, there are many comparative constructions of the kind ... more (NNS) than ..., as in IBM sold more computers than last year. Such constructions, and others of the same type, cannot be disambiguated by head-lexicalized grammars since they lexicalize the constituent with computers and not with more. It should be clear that a DOP1 model of depth 3 (subtrees of maximal depth 3) can already handle all these fine-grained distinctions. Bod (op cit.) points out other constructions containing dependencies involving non-head words. He also notes (personal communication) that one such construction involves only the word the, in that an NP which is lexicalized with the determiner the has a much higher chance of having a sister PP in the WSJ than an NP which is not lexicalized with the.

DOP can also be considered an instance of Machine Learning, in that it uses past experience to model new events. Other instances of learning include Memory-Based Learning (Stanfill & Waltz, 1986), Case-Based Reasoning (also known as Analogy-Based Reasoning: Aamodt & Plaza, 1994), Instance-Based Learning (Aha et al., 1991), and Similarity-Based Learning (Daelemans et al., 1996). Sima’an (1999:9) points out that, interestingly, “DOP extends the memory-based approach in two important ways: the analysis of new

---

6In this case the final definition of an elementary tree for TIGDOP given on the previous page changes from “all the daughter nodes of a left- or rightmost descendant” to “all the daughter nodes of the left- or rightmost descendant” in TIGDOP2.
input does not rely on a flat representation of the past-analyses but on a hierarchical – possibly recursive – linguistic structure of the analyses, and the analogy function is a probability function”.

2.2 Other DOP models: DOP2 and Beyond

DOP1 cannot parse sentences containing unknown words, as the recombination process can only use fragments contained in the corpora. Recall that DOP is a theory of both perception and production: it is clear that humans are able to employ strategies to deal with utterances containing words they may never have heard before. It is with this in mind that Bod (1995; 1998) constructs new DOP models.

2.2.1 DOP2: Dealing with Unknown Words

One simple strategy when encountering a new word would be to assign all possible categories to this new word and select the most probable parse from the resultant candidates along the lines of Tree-DOP. Bod (1998:70) deems this the partial parse method.

Nevertheless, he points out that such a method is statistically unsound, in that it does not provide probabilities for the given sentence, but rather the ‘sentence’ without the unknown word(s). Of course, once we know what the correct parse is, this will be added to a DOP treebank for later processing, i.e. we can never “forget” its meaning. However, if DOP is a psychologically plausible theory of language perception, we must be able to factor in the ‘ability’ to forget.

Results with DOP2 on experiments restricting fragments to depth 3 show that as with DOP1 (ignoring unknown words), parse accuracy improves as the maximum depth of fragments permitted increases. DOP1 would have a 0% parse accuracy for any sentence containing an unknown word. The results with DOP2 are significantly different for sentences containing an unknown word (42%) compared with parse accuracy for sentences containing known words (73%), for the same fragment depth.

Bod (op. cit.) examines some instances where DOP2 is led astray. For example, the phrase servicing Boston is analysed as a PP where servicing is the unknown word, as there are many instances in the ATIS corpus of the sequence pp[IN np[mnp[Boston]]. Bod points out that whilst such an analysis is perhaps merited on semantic grounds, our knowledge that servicing is connected to the verb service precludes such an analysis. Another example is where the fewest number of passengers is misanalysed as a noun-noun compound (where fewest is unknown), owing to numerous occurrences of strings such as ‘the flight number of...’.

A rather more interesting example is (54), where movies is unknown:

\[
\text{(54)} \quad \text{What movies are scheduled for these flights?}
\]

DOP2 misanalyses movies as a singular noun NN rather than a plural noun NNS. At first glance, this is surprising, given that there are many subtrees of category SBARQ in the training corpus with correct subject-verb agreement. DOP prefers parses which can be constructed using the largest possible fragments, so we might expect it to get such examples right. However, in this case the category SBARQ is significantly
outnumbered by sequences containing the category WHNP so that the most probable parse was constructed using this subtree, rather than the correct one.

What is significant about DOP2’s misanalysis of (54) is that the word *movies* is not unknown; rather, it is unknown in this context, i.e. it has been seen before, but with a different tag. Bod calls these unknown category words. One possibility in this case is to allow all words to have any lexical category, and let DOP2 produce the most probable parse in each case. Of course, the amount of processing for sentences of any sizeable length increases by a huge factor, and we can anticipate that DOP2 will produce many wrong parses—if it gets 42% right on sentences where there are very few unknown words, then we can expect a severe degradation in performance where all words are treated as potentially unknown. Furthermore, as Bod points out (op cit., p.79), “every sentence of the same length will get the same most probable parse”. Despite the small size of the ATIS corpus, this problem will recur in any treebank: no corpus will contain all words with all their possible categories.

2.2.2 DOP3: an Adaptive Dynamic System

One of the reasons the partial parse method does not work well may be due to the fact that DOP2 does not factor the unknown category words into its probability calculations. Again, it is clear that humans do have expectations about sentence structures they encounter which may be impossible to construct purely on the basis of known subtrees, i.e. we must consider our knowledge of language as incomplete. If DOP is to be considered a theory of perception, then it must be able to derive unknown subtrees as well as estimating their probabilities.

Of course, neither unknown subtrees nor their probabilities can be derived in advance. While it is easy to see which words in a sentence may be unknown, it is non-trivial to establish which words are unknown with respect to a category, as all words are potentially unknown category words. With respect to their probabilities, merely counting their relative frequencies, as in Tree-DOP, is no longer an option.

No treebank, no matter how large, can be considered to be complete; rather, it should be thought of as a sample of a total population. An unknown subtree may be considered as having a non-zero probability in this total population. Furthermore, observed subtrees may have different probabilities than their relative frequency in a corpus suggests. Therefore, the observed probabilities of fragments need to be adjusted to take into account the fact that their actual probabilities may differ in the total population. Bod (1998) uses the Good-Turing method (Good 1953) to adjust the frequencies of the observations (and hence their probabilities) in this way. This method is largely independent of the probability distribution, as it is assumed that the sample is obtained at random from the total population.

The Good-Turing Method

$r$ is the observed frequency of a type $t$, and $r^*$ is its adjusted frequency. So as to estimate $r^*$, Good-Turing defines $n_r$ as the number of types which occur $r$ times in a given sample, i.e. the frequency of $r$, as in (55), assuming the entire distribution \{n_1,n_2...n_t\} is available:
<table>
<thead>
<tr>
<th>( r )</th>
<th>( n_r )</th>
<th>( r^* )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>83000000000</td>
<td>0.0000073</td>
</tr>
<tr>
<td>1</td>
<td>60416</td>
<td>0.30</td>
</tr>
<tr>
<td>2</td>
<td>9057</td>
<td>1.37</td>
</tr>
<tr>
<td>3</td>
<td>4161</td>
<td>1.86</td>
</tr>
<tr>
<td>4</td>
<td>1944</td>
<td>1.99</td>
</tr>
<tr>
<td>5</td>
<td>773</td>
<td>3.74</td>
</tr>
<tr>
<td>6</td>
<td>482</td>
<td>4.37</td>
</tr>
</tbody>
</table>

Table 2.2: Adjusted Frequencies for NP-fragments from the ATIS corpus

(55) \[ r^* = (r + 1) \frac{n_{r+1}}{n_r} \]

Now the estimated probability of a type can be estimated by \( r^*/N \), where \( N \) is the total number of types observed. When \( n_r \) is large, good estimates for \( r^*/N \) can be obtained: it tends to be so for small frequencies \( r \), whereas if \( n_r \) is small, then \( r \) is usually so large that it needs no adjustment.

When it comes to adjusting the frequencies of an unseen type, \( r^* = n_1/n_0 \) where \( r = 0 \), and where \( n_0 \) equals the number of types we have not seen. That is, \( n_0 \) is the difference between the total number of types and the number of observed types. It follows that in order to calculate the probability of an unseen type, we need to know the total number of types in the population. The proportion of the population represented by the sample is shown to be \( 1 - n_1/N \) (Bod, 1998). Finally, it should be noted that the adjusted frequencies (and associated probabilities) of all unseen types are (usually) identical. That is, Good-Turing makes no distinction between types we have not seen and those that we have (although more sophisticated methods exist, such as linear interpolation and backoff models (Manning & Schütze, 1999:217–220)).

In DOP models, subtrees of each category constitute sub-classes in the probability mass (each summing to 1). Good-Turing must, therefore, be applied to each sub-class in turn, i.e. the probabilities of the NP-fragments will be adjusted, as will the VP-fragments, the S-fragments, and so on.
Bod (1998:86-87) gives an illustration of Good-Turing in operation on the 118348 NP-fragments from the ATIS corpus, and we reproduce that table here as Table 2.2. To reiterate, \( r \) depicts the number of NP-subtrees observed from 0-6, \( n_r \) shows the number of NPs having that frequency in the corpus, and \( r^* \) shows the adjusted frequencies via Good-Turing. To explain this formula with respect to these numbers, let us take \( r = 3 \). In this case, \( n_r = 4161 \), and \( n_{r+1} = 1944 \). Therefore, according to formula (55), \( r^* = (3+1) \times 1944/4161 = 1.86 \). The numbers for \( r = 0 \) depend on estimating \( n_0 \), that is, how many NP-fragments we have yet to see, so we need to estimate the total population of NPs. We may use Good’s approximation formula \( 1 - n_1/N \), but in DOP models, any terminal fragment can mismatch with any word of the input, so using Good’s formula here would considerably underestimate the potential number of NP fragments. Bod (1998) proposes to use the approximation formula to estimate the total number of words in the ATIS corpus, from which the number of possible NP-fragments can be calculated by deriving the number of possible attachments of words in the ATIS corpus to the NP-fragments containing no terminal vocabulary. As an example, for every NP-subtree in the corpus of the type shown in (56), it is assumed that any word in the ATIS corpus can be both a Det and an N, given that all words are presumed to be unknown with respect to category:

\[
\begin{array}{c}
\text{NP} \\
\text{Det} \quad \text{N}
\end{array}
\]

(56)

We can now show how the numbers for \( n_0 \) in Table 2.2 are arrived at:

1. there are 1351 word types in the ATIS corpus, 307 of which occur once. The sample/population ratio is approximated as \( 1 - 307/1351 = 0.773 \). Therefore, the approximate number of distinct word types in ATIS is \( 1351/0.773 = 1750 \).

2. we assume that each of these 1750 word types can occur as any category in any NP-subtree containing vacant lexical items, to which we must add the number of distinct non-lexical NP-fragments, which together yields \( 8.3 \times 10^6 \), i.e. \( n_0 \) in Table 2.2.

3. the number of unseen NP-fragments \( n_0 \) is equal to the number of distinct NP-types \( 8.3 \times 10^6 \) minus the observed number of distinct NP-fragments, in this case 77479. This approximates to \( 8.3 \times 10^6 \), showing that given the wide-ranging powers of the mismatching allowed, hardly any of the possible NP-fragments have been seen so far.

4. the adjusted frequency for unseen NP-fragments, via Good-Turing, is \( n_1/n_0 = 60416/8.3\times10^6 = 0.0000073 \), i.e. \( r^* \) in Table 2.2. What this means in practice is that for each of the approximately \( 8.3\times10^6 \) unseen NP-types, we are assuming that we have seen them 0.0000073 times, rather than zero times.

Of course, this latter figure is extremely small given the decision to treat all words as unknown-category words. If this were to be restricted, the number of unseen NP-fragments decreases, which means that their adjusted frequency would increase.

Note that other methods can be used to deal with the problem of unknown words. For instance, Sima’an (1999:139) observes that “for enabling the parsing and disambiguation of sentences containing words that
are unknown to the parser, we employ a simplified version of the Add-One method (Gale & Church, 1994) ... [which] is of course a wrong assumption that results in assigning too small probabilities to unknown words. The only reason for making this assumption is the simplicity of its implementation”.
We show in section 6.3.6 that following (Bod, 2000a, we intend to use Good-Turing to limit the scope of the LFG-DOP Discard operator, as well as using it to help deal with unknown words. As it is already being used to adjust probabilities of known events, integrating its use for this purpose also should prove unproblematic.

Experiments with DOP3

We have reported that the estimated number of possible NP-fragments for the ATIS domain is about 12000 times the actual set of observed NP-fragments. In order to run experiments in a practical amount of time, the number of mismatches will need to be limited. For the ATIS domain, Bod (1998:90) limits mismatches to nouns and verbs, these being the only lexically ambiguous classes in his corpus.
For sentences containing both unknown words and unknown category words, Bod claims a parse accuracy of 62%, and for sentences containing just unknown category words, a parse accuracy of 86%. Recall that on the same data, DOP2’s parse accuracy for the former class of sentences was 42%. For Bod’s small test set, this represents an increase of 5 correctly parsed sentences out of 26. He notes that 3 of these 5 sentences contain unknown category words, which DOP2 cannot handle. The other two show how DOP3 copes with unknown words. Using (54), p.56, as an example, DOP3 correctly tags *movies* as a plural noun (NNS), where DOP2 tagged it as a singular noun (NN), owing to an un-DOP-like bias for smaller trees. DOP’s normal propensity for preferring larger trees returns in the DOP3 model.

2.2.3 Some General Properties of DOP Models

Bod (*op cit.*, pp.64-65) lists a number of properties pertaining to Tree-DOP models. All of these concern the effect on parse accuracy if variations are made to the DOP models used. For instance, any (or all) of the following might be discarded:

- once occurring fragments;
- fragments of depth greater than three;
- fragments containing non-head words.

If any subset of fragments are removed from the treebank, the results achieved will be adversely affected.
Having run a number of experiments, some other general traits of DOP models have been ascertained which, to our knowledge have not been documented. We consider them to be of interest, especially where some of the characteristics might be seen as counter-intuitive.
Probability of Sentences

Should clarity on this issue be needed, we provide results from three small experiments which show definitively that the probability of a sentence is not equal to the probability of the subject NP multiplied by the probability of the VP. This is the case for a zero-order Markov model, as in (57):

\[
(57) \quad P((NP,VP)) = P(NP).P(VP)
\]

It is not the case for other models, where in order to calculate the probability of a sentence given a model, \( P(S \mid M) \), the model \( M \) is always a parameter to be factored in. Thus for a 1st-order Markov model and for a PCFG, the calculations are as in (58):

\[
(58) \quad \text{1st-order Markov: } P((NP,VP)) = P(NP).P(VP \mid NP)
\]

\[
\text{PCFG: } P((NP,VP)) = \sum_{Trees \ \text{generating } (NP,VP)} P(Trees, (NP,VP))
\]

In the following experiments, the model \( M \) is Tree-DOP. Assuming three DOP treebanks built from the first three, four and five sentences in (59), the results are shown in Table 2.3:

\[
(59) \quad \begin{align*}
\text{a.} & \quad \text{Marie se suicide.} \\
\text{b.} & \quad \text{Jean commet un crime.} \\
\text{c.} & \quad \text{Le suicide est tragique.} \\
\text{d.} & \quad \text{Jean commet le meurtre.} \\
\text{e.} & \quad \text{Marie commet un attentat.}
\end{align*}
\]

These sentences were designed to test the impact of the wrong, compositional translation (i.e. \textit{commet le suicide}) on the correct, specific translation (i.e. \textit{se suicide}). We comment further on such experiments in section 3.2.2 (cf. (112)–(116)), section 5.2.3 (cf. (334)–(342)), and section 6.1.1 (cf. (362)–(363)).

In column 4 in Table 2.3, there is obviously no factoring in of the probability of the sentence itself, plus other fragments into which either subject NP or VP can be substituted to produce a derivation which will comprise part of the overall probability of the sentence in question. Recall that the probability of a sentence is computed by \textit{summing} all the derivations which can produce the particular analysis, as in (27), p.36. We comment further on the probabilities of the NPs as subject (column 2 in Table 2.3) in the following section.

Calculating the Probability of NP Subjects in DOP

Let us assume that a tree such as (60) is derived from a treebank containing only intransitive or transitive verbs:

61
<table>
<thead>
<tr>
<th>Sentence</th>
<th>P(NP=SUBJ)</th>
<th>P(VP)</th>
<th>P(NP=SUBJ)*P(VP)</th>
<th>P(S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marie se suicide</td>
<td>0.0925</td>
<td>0.0833</td>
<td>0.0077</td>
<td>0.0333</td>
</tr>
<tr>
<td>Jean se suicide</td>
<td>0.2434</td>
<td>0.0833</td>
<td>0.0203</td>
<td>0.0091</td>
</tr>
<tr>
<td>Jean commet le suicide</td>
<td>0.2434</td>
<td>0.0417</td>
<td>0.0101</td>
<td>0.0161</td>
</tr>
<tr>
<td>Marie commet le suicide</td>
<td>0.0925</td>
<td>0.0417</td>
<td>0.0039</td>
<td>0.0018</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sentence</th>
<th>P(NP=SUBJ)</th>
<th>P(VP)</th>
<th>P(NP=SUBJ)*P(VP)</th>
<th>P(S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marie se suicide</td>
<td>0.0667</td>
<td>0.0481</td>
<td>0.0032</td>
<td>0.0185</td>
</tr>
<tr>
<td>Jean se suicide</td>
<td>0.3467</td>
<td>0.0481</td>
<td>0.0167</td>
<td>0.0038</td>
</tr>
<tr>
<td>Jean commet le suicide</td>
<td>0.3467</td>
<td>0.0808</td>
<td>0.0280</td>
<td>0.0385</td>
</tr>
<tr>
<td>Marie commet le suicide</td>
<td>0.0667</td>
<td>0.0808</td>
<td>0.0054</td>
<td>0.0045</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sentence</th>
<th>P(NP=SUBJ)</th>
<th>P(VP)</th>
<th>P(NP=SUBJ)*P(VP)</th>
<th>P(S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marie se suicide</td>
<td>0.1869</td>
<td>0.0333</td>
<td>0.0062</td>
<td>0.0145</td>
</tr>
<tr>
<td>Jean se suicide</td>
<td>0.2691</td>
<td>0.0333</td>
<td>0.0090</td>
<td>0.0021</td>
</tr>
<tr>
<td>Jean commet le suicide</td>
<td>0.2691</td>
<td>0.0126</td>
<td>0.0280</td>
<td>0.0175</td>
</tr>
<tr>
<td>Marie commet le suicide</td>
<td>0.1869</td>
<td>0.0087</td>
<td>0.0054</td>
<td>0.0064</td>
</tr>
</tbody>
</table>

Table 2.3: P(S) ≠ P(NP=SUBJ)*P(VP)
(60)  

S

NP  VP

The question is as follows: given a VP derived randomly from the treebank, which is more likely as subject of that VP—John, which appears in the corpus as subject of transitive verbs only, or Mary, which appears in the corpus as subject of intransitive verbs only? Let Hypothesis 1 be that Mary is more likely, while Hypothesis 2 states that John is more likely. In order to try and skew the results as far in favour of Hypothesis 1 as possible, let us assume the corpus in (61):

(61)  
a. John loves Sally.  
b. Sally laughs.  
c. Mary cries.  

That is, there are two intransitive verbs in (61), and we also simplify the object NP to cut down as much as possible the proliferation of s(np(john)...) fragments in the treebank. Given this, one might expect that Mary, being subject of an intransitive verb in the corpus, would be more likely as an overall subject than John, there being twice as many instances in the treebank into which it could fit. However, we shall show that despite this inherent bias in the corpus, John is the preferred subject.

Assuming the corpus in (61), we derive the treebank in (62):
<table>
<thead>
<tr>
<th>Sentences</th>
<th>Verbphrases</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: ( s(np(_), vp(_)) ).</td>
<td>1: ( vp(v(_)) ).</td>
</tr>
<tr>
<td>2: ( s(np(_), vp(_)) ).</td>
<td>2: ( vp(v(_)) ).</td>
</tr>
<tr>
<td>3: ( s(np(_), vp(_)) ).</td>
<td>3: ( vp(v(_), np(_)) ).</td>
</tr>
<tr>
<td>4: ( s(np(_), vp(v(_))) ).</td>
<td>4: ( vp(v(_), np(sally)) ).</td>
</tr>
<tr>
<td>5: ( s(np(_), vp(v(_))) ).</td>
<td>5: ( vp(v(loves), np(_)) ).</td>
</tr>
<tr>
<td>6: ( s(np(_), vp(v(_), np(_))) ).</td>
<td>6: ( vp(v(loves), np(sally)) ).</td>
</tr>
<tr>
<td>7: ( s(np(_), vp(v(loves), np(sally))) ).</td>
<td>7: ( vp(v(cries)) ).</td>
</tr>
<tr>
<td>8: ( s(np(_), vp(v(loves), np(sally))) ).</td>
<td>8: ( vp(v(laughs)) ).</td>
</tr>
<tr>
<td>9: ( s(np(_), vp(v(cries))) ).</td>
<td></td>
</tr>
<tr>
<td>(62)</td>
<td>Verbs</td>
</tr>
<tr>
<td>10: ( s(np(_), vp(v(laughs))) ).</td>
<td></td>
</tr>
<tr>
<td>11: ( s(np(john), vp(_)) ).</td>
<td></td>
</tr>
<tr>
<td>12: ( s(np(john), vp(v(_), np(_))) ).</td>
<td>1: ( v(cries) ).</td>
</tr>
<tr>
<td>13: ( s(np(john), vp(v(_), np(sally))) ).</td>
<td>2: ( v(laughs) ).</td>
</tr>
<tr>
<td>14: ( s(np(john), vp(v(loves), np(_))) ).</td>
<td>3: ( v(loves) ).</td>
</tr>
<tr>
<td>15: ( s(np(john), vp(v(loves), np(sally))) ).</td>
<td></td>
</tr>
<tr>
<td>16: ( s(np(mary), vp(_)) ).</td>
<td>Nounphrases</td>
</tr>
<tr>
<td>17: ( s(np(mary), vp(v(_))) ).</td>
<td></td>
</tr>
<tr>
<td>18: ( s(np(mary), vp(v(cries))) ).</td>
<td>1: ( np(john) ).</td>
</tr>
<tr>
<td>19: ( s(np(sally), vp(_)) ).</td>
<td>2: ( np(mary) ).</td>
</tr>
<tr>
<td>20: ( s(np(sally), vp(v(_))) ).</td>
<td>3: ( np(sally) ).</td>
</tr>
<tr>
<td>21: ( s(np(sally), vp(v(laughs))) ).</td>
<td>4: ( np(sally) ).</td>
</tr>
</tbody>
</table>

The probability of an NP being the subject of any sentence in the treebank is equal to the number of sentences with an uninstantiated NP times that NP, plus the number of sentences already containing that NP, divided by all sentence fragments in the treebanks. That is, the probability of NP(SUBJ = John) is shown in (63):

\[
\text{P(NP-SUBJ = John)} = \frac{10 + \frac{2}{21} + \frac{2}{21}}{21} = \frac{30}{64}.
\]

The three possible NP subjects have the probabilities in (64):

\[
\text{(64) a. P(SUBJ = John)} = \frac{30}{64} = 0.357
\]

\[
\text{b. P(SUBJ = Mary)} = \frac{10 + \frac{1}{21} + \frac{2}{21}}{21} = \frac{22}{64} = 0.262
\]

\[
\text{c. P(SUBJ = Sally)} = \frac{10 + \frac{2}{21} + \frac{4}{21}}{21} = \frac{32}{64} = 0.381
\]

If the object NP had been something other than Sally, then P(SUBJ = Sally) = P(SUBJ = Mary), with the remaining \( \frac{10}{64} \) of the probability space occupied by this new NP, (say) Anne.

However, an objection to the above calculations is that they mean very little, when what we want instead is to see how likely an NP is as subject of a particular type of NP, either transitive or intransitive. That is, to
decide in favour of one hypothesis or the other, we want to know which is bigger, \( P(\text{Mary} \mid VP_{\text{trans}}) + P(\text{Mary} \mid VP_{\text{intr}}) \), or \( P(\text{John} \mid VP_{\text{trans}}) + P(\text{John} \mid VP_{\text{intr}}) \)? Whilst at first glance this seems reasonable, it is a little too simplistic, as the probabilities of the sentences themselves do not factor into these calculations (we know that \( P(S) \neq P(NP).P(VP) \) from the last section). Therefore, to compute these numbers we shall have to calculate the probabilities of each sentence, summing each \( NP + VP_{\text{intr/\text{trans}} } \) sequence, and adding that to the value of that \( NP + VP_{\text{intr/\text{trans}} } \) sequence.

How many sentences are there? Just counting distinct sentences (we do not need to worry which Sally we take to occupy subject and object positions), there are nine strings possible with ‘object-less’ verbs (as opposed to proper intransitives), as shown in (65):

\[
S \\
\downarrow \\
NP \quad VP \\
\downarrow \\
V \\
3 \quad 3
\]

(65)

There are 27 strings possible with objects, as shown in (66):

\[
S \\
\downarrow \\
NP \quad VP \\
\downarrow \\
V \quad NP \\
3 \quad 3 \quad 3
\]

(66)

We see, therefore, that we are three times more likely to encounter a sentence with a ‘transitive’ VP (i.e. one containing an object, whether this be grammatical or not) than we are to see an ‘intransitive’ VP (i.e. one containing no object). It is this fact that sways the argument in favour of John. Note that if we assume, as before, a fourth NP Anne appearing as object of loves, then we obtain 12 intransitive sentences and 48 transitives, in which case transitives become four times more likely. Let us now calculate the probabilities of all 36 sentences. The probabilities of the intransitive sentences are as in (67):

\[
\begin{align*}
P(\text{John cries}) &= 75/2016 \\
P(\text{Mary cries}) &= 203/2016 \\
P(\text{Sally cries}) &= 162/2016 \\
P(\text{John loves}) &= 30/2016 \\
P(\text{Mary loves}) &= 62/2016 \\
P(\text{Sally loves}) &= 84/2016 \\
P(\text{John laughs}) &= 75/2016 \\
P(\text{Mary laughs}) &= 107/2016 \\
P(\text{Sally laughs}) &= 258/2016
\end{align*}
\]

(67)

This totals 1056/2016, meaning that intransitive sentences occupy 52.4% of the probability space. As an example, consider \( P(\text{John loves}) \). The possible derivations of this sentence given the treebank in (62) are those in (68):
(68) \[ s_{11 \cdot np} = \frac{1}{2\pi} \cdot \frac{1}{2} = \frac{1}{2\pi} \]
\[ s_{4 \cdot np} = \frac{1}{2\pi} \cdot \frac{1}{2} = \frac{1}{2\pi} \]
\[ s_{1 \cdot np \cdot np} = \frac{3}{4\pi} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{16\pi} \]

Summing the probabilities of the derivations in (68) gives \( P(\text{John loves}) = 15/1008 \) (or 30/2016, as in (67)).

The probabilities of the transitive sentences are as in (69):

\[
\begin{align*}
P(\text{John loves John}) &= 47/2016 \quad P(\text{Mary loves John}) = 15/2016 \\
P(\text{Sally loves John}) &= 26/2016 \quad P(\text{John loves Mary}) = 47/2016 \\
P(\text{Mary loves Mary}) &= 15/2016 \quad P(\text{Sally loves Mary}) = 26/2016 \\
P(\text{John loves Sally}) &= 258/2016 \quad P(\text{Mary loves Sally}) = 66/2016 \\
P(\text{Sally loves Sally}) &= 108/2016 \quad P(\text{John cries Sally}) = 70.5/2016 \\
P(\text{Mary cries Sally}) &= 22.5/2016 \quad P(\text{Sally cries Sally}) = 39/2016 \\
P(\text{John laughs Sally}) &= 70.5/2016 \quad P(\text{Mary laughs Sally}) = 22.5/2016 \\
P(\text{Sally laughs Sally}) &= 39/2016 \quad P(\text{John cries John}) = 11.75/2016 \\
P(\text{Mary cries John}) &= 3.75/2016 \quad P(\text{Sally cries John}) = 6.5/2016 \\
P(\text{John cries Mary}) &= 11.75/2016 \quad P(\text{Mary cries Mary}) = 3.75/2016 \\
P(\text{Sally cries Mary}) &= 6.5/2016 \quad P(\text{John laughs John}) = 11.75/2016 \\
P(\text{Mary laughs John}) &= 3.75/2016 \quad P(\text{Sally laughs John}) = 6.5/2016 \\
P(\text{John laughs Mary}) &= 11.75/2016 \quad P(\text{Mary laughs Mary}) = 3.75/2016 \\
P(\text{Sally laughs Mary}) &= 6.5/2016
\end{align*}
\]

This totals \( 960/2016 \), meaning that transitive sentences occupy 47.6\% of the probability space. Given these probabilities, we can now make our final calculations:

- \( P(\text{Mary} \mid V_{\text{intrans}}) = \)
  - \( P(\text{Mary} \mid \text{cries}) = 203/2016 \)
  - \( P(\text{Mary} \mid \text{laughs}) = 107/2016 \)
  - \( P(\text{Mary} \mid \text{loves}) = 62/2016 \)
  - \( = 372/2016 \)

- \( P(\text{John} \mid V_{\text{intrans}}) = \)
  - \( P(\text{John} \mid \text{cries}) = 75/2016 \)
  - \( P(\text{John} \mid \text{laughs}) = 75/2016 \)
  - \( P(\text{John} \mid \text{loves}) = 30/2016 \)
  - \( = 180/2016 \)

That is, \( \text{Mary} \) is more than twice as likely to be the subject of an intransitive verb than \( \text{John} \) is.
\bullet P(\text{Mary} \mid V_{trans}) = 156/2016 \\
\bullet P(\text{John} \mid V_{trans}) = 540/2016 \\

That is, \textit{John} is about 3.5 times as likely to be the subject of a transitive verb than \textit{Mary} is. Adding our numbers, we now see that:

\begin{itemize}
  \item \(P(\text{Mary} \mid V_{trans}) + P(\text{Mary} \mid V_{intrans}) = \frac{528}{2016} = 0.262\)
  \item \(P(\text{John} \mid V_{trans}) + P(\text{John} \mid V_{intrans}) = \frac{720}{2016} = 0.357\)
\end{itemize}

leaving \(P(\text{SUBJ} = \text{Sally}) = 0.381\). Again, if the object NP in (61a) had been something other than \textit{Sally}, then \(P(\text{SUBJ} = \text{Sally}) = P(\text{SUBJ} = \text{Mary})\), with the remaining 10/84 (0.119) of the probability space occupied by this new NP (\textit{Anne}, say).

A number of conclusions present themselves:

1. As expected, different NPs are more likely as subjects of different types of verb, heavily biased by the type of verb they occur with in the corpus.

2. The bigger the tree in which an element is contained in the corpus, the more likely that element is compared to elements of the same class (NP, say).

3. Calculating the probabilities of subject NPs \textit{post hoc} does not alter the figures at all.

4. Hypothesis 2 is correct. The number of possible strings in which \textit{John} can occur outweighs the inherent bias in the corpus (twice as many intransitives as transitives), rendering this NP more likely as a subject (given this corpus) than \textit{Mary}.

\textbf{Properties of Redundancy in DOP}

As we have just shown with respect to subjects, Bonnema \textit{et al.} (2000) similarly observe that the notion of redundancy, which is critical if relative frequency is the criterion by which probabilities of fragments are to be calculated, can result in undesirable statistical biases. As a result they propose an alternative probability model which we describe in section 6.3.5.

Bonnema \textit{et al. (op cit.)} state that “the most important innovation that the data oriented approach has brought to stochastic parsing is the decision to use \textit{all} fragments from the corpus”. Whilst this is not strictly correct (fragments can be removed from the probability space on the grounds of their depth, or the number of terminals they contain, for instance), it is indeed one possible DOP model. In such cases, they note that “the exponential nature of the fragment extraction operation implies that large corpus trees make a disproportionately large contribution to the probability mass of the fragments”. We shall repeat their Figure 1 here as (70) to exemplify the problem:
Intuitively, the preferred analysis for the string \( ab \) should be (70b). This is not, however, the preferred analysis given the traditional model of Tree-DOP. Let \( \rho(\alpha) = \beta \) indicate that tree with root \( \alpha \) has \( \beta \) subtrees. Given (70), \( \rho(A) = 134 \), so all fragments with root \( A \) have probability \( \frac{1}{134} \), including (70b). Contrast this with \( \rho(X) = 4 \). We shall show in (71) and (72) that \( X (Y(a) Z(b)) \) gets selected as the preferred analysis of \( ab \).

\[
\begin{align*}
\text{(71)} & \quad X \quad \circ \quad Y \quad \circ \quad Z \quad = \frac{1}{4} \cdot \frac{1}{2} = \frac{1}{8} \\
\text{(72)} & \quad B \quad C \quad = \frac{1}{134}
\end{align*}
\]

That is, \( P(X (Y(a) Z(b)) = 1/16 \), there being only one possible derivation given fragments obtained from the trees in (70). The derivations of \( P(A (B(a) C(b))) \) are those in (72), assuming the same set of fragments:

\[
\begin{align*}
\text{(72)} & \quad B \quad C \quad = \frac{1}{134} \\
\text{(72)} & \quad B \quad C \quad = \frac{1}{134} \cdot 1 = \frac{1}{134} \\
\text{(72)} & \quad B \quad C \quad = \frac{1}{134} \cdot 1 = \frac{1}{134} \\
\text{(72)} & \quad B \quad C \quad = \frac{1}{134} \cdot 1 = \frac{1}{134}
\end{align*}
\]

Given (27), the probability of a parse-tree is computed by summing all its derivations, so \( P(A (B(a) C(b))) = \frac{4}{134} \), or \( \frac{1}{33.5} \). Thus it can be seen that despite there being four possible derivations of \( A (B(a) C(b)) \), its probability is slightly less than half that of \( P(X (Y(a) Z(b)) \).
Bonnema et al. propose to deal with such undesirable results by imposing constraints on what fragments can contribute to derivations of particular analyses. For example, limiting the number of possible substitution sites in a tree fragment will disqualify most of the fragments derivable from the rightmost tree in (70), while at the same time preserving most (if not all) of the smaller fragments. In so doing, the relative frequencies of subtrees and hence the probabilities of derivations will increase, thereby shifting the probabilities in favour of analyses from the smaller trees. The essence of the problem, as Bonnema et al. point out, is that

“the substitution probability of a fragment (relative to all fragments of the same category) is not proportional to the relative occurrence frequency of the fragment in the treebank. The classical DOP model thus employs a probability measure that invalidates the principle of preferring frequently occurring structures over alternatives that occur less frequent (sic), and should therefore be abolished”.

While this is true, Tree-DOP is the simplest model possible, despite being based on the incorrect assumption that all fragments are independent. Bonnema et al. subsequently propose a new probability model to redress this failing. We describe this model further in section 6.3.5.
Chapter 3

Translation using LFG

LFG (Kaplan & Bresnan, 1982) is a theory of syntax which allows description of sentences at a number of different levels—c-structure, f-structure and s-structure. The c-structure (constituent structure) is a phrase-structure tree signifying the surface structure of the string, the f-structure (functional structure) is an attribute-value matrix (AVM) capturing the grammatical relations inherent in the string, while s-structures (semantic structure) go one level deeper and express relational information. There are two mappings: φ, which maps c-structure nodes onto elements of the f-structure, and σ, which relates f- and s-structures, so that LFG is a linear model (c → f → s-structure). An example of the three structures for the sentence *John swims* is shown in (73):

\[
\begin{array}{c}
S \\
\downarrow \\
NP \downarrow \\
John \\
\downarrow \\
VP \downarrow \\
V \downarrow \\
\text{swims}
\end{array}
\rightarrow
\phi
\begin{array}{c}
\text{SUBJ} \quad \text{PRED} \quad \text{NUM} \quad \text{SG} \\
\text{PRED} \quad \text{swim}((\uparrow\text{SUBJ}))' \\
\text{TENSE} \quad \text{PRES}
\end{array}
\rightarrow
\sigma
\begin{array}{c}
\text{REL} \quad \text{SWIM} \\
\text{ARG1} \quad \text{JOHN}
\end{array}
\]

LFG has also been proposed as a formalism for MT (Kaplan et al., 1989). Given the very powerful and elegant way of relating unlike representations (c-structure trees, and f- and s-structure AVMs) when used as a pure linguistics theory of syntax, it is not too surprising that its use as an MT engine has met with some success. Kaplan et al. illustrate the ability of LFG to cope with some hard examples using codescription, i.e. various levels of linguistic structure contribute to (‘co-describe’) the translation of strings. Rather than conflating all translationally relevant information into a single, linguistically hybrid level of representation, LFG-MT allows information from different linguistic levels of representation to interact in order to constrain the translation relation, by function composition.

The original model of Kaplan et al. (*op cit.*) introduces the r-correspondence as a mapping between source and target f-structures, and the r’-correspondence as a mapping between source and target s-structures. The general architecture for LFG-MT is shown in (74):
These mappings allow the formation of such descriptions as those in (75):

(75) a. $\tau(\uparrow \text{COMP}) = (\tau \uparrow \text{XCOMP})$

b. $\tau'\sigma(\uparrow \text{SUBJ}) = \sigma(\tau \uparrow \text{TOPIC})$

The first of these maps a source $f$-structure COMP (sentential object) to a target $f$-structure XCOMP (infinitival object), and the second imposes the constraint that the semantics of the source SUBJ translates into the semantics of the target TOPIC via $\tau'$, without stipulating what the contents of those $\sigma$-structures should be.

There are other constraints playing a part in the formation of the translation which are not imposed by the source structures. For example, translating from English into French would not require information about gender to be part of the translation equations; rather, and quite properly, such information is completely specified in the target grammar and lexicon. Given this, it can be seen that the description of the target structures derived from their source equivalents will often be incomplete. Kaplan et al. (op cit., p.276) state, therefore, that:

“... for a target sentence to be an adequate translation of a given source sentence, it must be the case that a minimal structure assigned to that sentence by the target grammar is subsumed by a minimal solution to the transfer description”.

To illustrate the translation process, Kaplan et al. use the French-German example (76):

(76) Der Student beantwortet die Frage $\leftrightarrow$ L’étudiant répond à la question.

Here the verbs in question differ as to what type of object they require: beantworten takes a direct object, whilst répondre requires an oblique (prepositional) object, as shown by their semantic forms in (77):

(77) a. $(\uparrow \text{PRED}) = \text{‘beantworten}(\uparrow \text{SUBJ}(\uparrow \text{OBJ}))’$

b. $(\uparrow \text{PRED}) = \text{‘répondre}(\uparrow \text{SUBJ}(\uparrow \text{OBL}))’$

The transfer lexicon for beantworten would be as in (78):
(78) beantworten:
 \((\tau \uparrow \text{PRED}) = \text{répondre}\)
 \((\tau \uparrow \text{SUBJ}) = \tau(\uparrow \text{SUBJ})\)
 \((\tau \uparrow \text{OBL OBJ}) = \tau(\uparrow \text{OBJ})\)

This states that \text{répondre} is its corresponding French predicate, that the translation of the SUBJ is straightforward, and that the translation of its OBJ is the OBL OBJ of \text{répondre}. Let us assume the source \(f\)-structure (79):

\[
\begin{array}{c|c}
\text{TENSE} & \text{PRED} \quad \text{SUBJ} \\
\text{PRES} & \quad \begin{array}{c|c}
\text{PRED} & \text{'Student'} \\
\text{NUM} & \text{SG} \\
\text{GEND} & \text{masc} \\
\text{SPEC} & \quad \begin{array}{c|c}
\text{DEF} & + \\
\text{PRED} & \text{der} \\
\end{array}
\end{array}
\end{array}
\]

(79)

\[
\begin{array}{c|c}
\text{TENSE} & \text{OBJ} \\
\text{PRES} & \quad \begin{array}{c|c}
\text{PRED} & \text{'Frage'} \\
\text{NUM} & \text{SG} \\
\text{GEND} & \text{fem} \\
\text{SPEC} & \quad \begin{array}{c|c}
\text{DEF} & + \\
\text{PRED} & \text{die} \\
\end{array}
\end{array}
\end{array}
\]

The \(\tau\)-equations in (78) (in addition to straightforward equations for NPs) will relate (79) into the target structure (80):

\[
\begin{array}{c|c}
\text{TENSE} & \text{OBJ} \\
\text{PRES} & \quad \begin{array}{c|c}
\text{PRED} & \text{'étudiant'} \\
\text{NUM} & \text{SG} \\
\text{GEND} & \text{masc} \\
\text{SPEC} & \quad \begin{array}{c|c}
\text{DEF} & + \\
\text{PRED} & \text{le} \\
\end{array}
\end{array}
\end{array}
\]

(80)

\[
\begin{array}{c|c}
\text{OBL} & \quad \begin{array}{c|c}
\text{PRED} & \text{'à([question])'} \\
\text{NUM} & \text{SG} \\
\text{GEND} & \text{fem} \\
\text{SPEC} & \quad \begin{array}{c|c}
\text{DEF} & + \\
\text{PRED} & \text{la} \\
\end{array}
\end{array}
\end{array}
\]

It is clear from this structure that the target grammar has added a significant amount of information not present in the translation relation in (78). This \(f\)-structure would constitute the input into the generation
process.

Kaplan et al. illustrate the ability of LFG-MT to cope with differences in control, as in (81):

\[(81) \quad \text{`likely}(\uparrow \text{XCOMP}) (\uparrow \text{SUBJ}) \iff \text{`probable}(\uparrow \text{COMP}) (\uparrow \text{SUBJ})\]

They also give examples of non-local dependencies, and differences in embedding, or “headswitching”. An example of the latter translation case is just \textit{\rightarrow} \textit{venir de}, where what is a dependant of node X in a source language becomes the governor of X in the target.

LFG-MT can also cope correctly with the \textit{like} \textit{\rightarrow} \textit{plaire} relation-changing case. Its solution is (82):

\[(82) \quad \text{like:}
\begin{align*}
(\tau \uparrow \text{PRED FN}) &= \text{plaire} \\
(\tau \uparrow \text{SUBJ}) &= (\tau \uparrow \text{OBL}) \\
(\tau \uparrow \text{OBJ}) &= (\tau \uparrow \text{SUBJ})
\end{align*}
\]

That is, the subject of \textit{like} is translated as the oblique argument of \textit{plaire}, while the object of \textit{like} is translated as the subject of \textit{plaire}.

In sum, Kaplan et al. view one of the main advantages of the codescription approach to translation as being the possibility of “expressing correspondences between separate pieces of linguistically motivated representations and in this way allowing the translator to exploit the linguistic descriptions of source and target language in a more direct way than is usually proposed” \textit{(op cit., p.281)}. 

### 3.1 On the Nature of Representations and Constraints

An interesting question concerns the difference between a representation and the constraints \textit{underlying} that representation. Prior to Johnson’s (1988) work on Attribute-Value Grammars, it is fair to say that many linguists had clouded this issue. Nevertheless, there is a clear distinction between linguistic objects and the language used to describe the constraints. For instance, Johnson observes that:

“In this system attribute-value structures play only one role; they are defined in order to give a semantics for the language that describes them. None of the algorithms in the following chapters actually constructs an attribute-value structure; instead, they operate on descriptions of attribute-value structures to determine the existence and the probabilities of those structures” \textit{(op cit., p.12)}.

For Johnson, therefore, the underlying linguistic constraints are what is important; the structures that can be built from them are merely an encoding of these constraints which make it easier to visualize what they refer to. He also notes with respect to LFG that:

“The success of the Kasper and Rounds (1986) and the Moshier and Rounds (1987) treatments of disjunction and negation suggests that the distinction between attribute-value structures and
their descriptions made by Kaplan & Bresnan (1982) and abandoned by Kay (1979) is an important one. Indeed, it is the fundamental distinction made in the treatment of attribute-value structures presented in the following chapters” (Johnson, 1988:11).

Also, for Johnson, “a linguistic structure of an attribute-value grammar is called an annotated constituent structure. It consists of a constituent structure tree, an attribute-value structure, and an association of the nodes in the constituent structure tree with elements of the attribute-value structure” (op cit., p.5). For him, then, it seems not to make sense to talk about the Attribute-Value structure itself in isolation of the c-structure and the mapping from c- to f-structure. This approach can be contrasted with Wedekind (1986, 1988) who uses an f-structure with associated constraints (about linear order—for Wedekind, c-structure does nothing more than encode linear precedence) for generation (cf. section 3.4).

3.2 A Problematic Case of Translation for LFG: Headswitching

Several problem cases for LFG-MT have been pointed out in the literature. We concentrate in this section on headswitching examples, but also report in subsequent sections on problems of scoping, translation of adjuncts and head-modifier constructions.

3.2.1 Embedded cases of Headswitching

The primary example used here by Kaplan et al. is the well-known headswitching example venir de X ←→ has just X-ed, as in (83):

(83) The baby just fell ←→ Le bébé vient de tomber.

They propose to deal with such problems in two ways. The first of these is as in (84):

(84) just:
(† PRED ) = ‘just(† ARG)’
(τ† PRED FN) = venir
(τ† XCOMP = τ(† ARG)

That is, the XCOMP function of venir (i.e. its infinitival complement: in (83), de tomber) corresponds to the ARG function (in (83), the baby fell) of just, as shown by the respective source and target f-structures in (85) and (86):

(85) \[
\begin{array}{c}
PRED \quad \text{‘just([fall])’} \\
\end{array}
\begin{array}{c}
| SUBJ \quad | PRED \quad \text{‘baby’} \\
| SPEC \quad | \quad \text{the} \\
| TENSE \quad | \quad \text{PAST} \\
| PRED \quad | \quad \text{‘fall([baby])’} \\
\end{array}
\]
The second approach is where just is not treated as a head subcategorizing for an ARG, but as a ‘normal’ adverbial sentential modifier. Instead, headswitching occurs between source and target f-structures, as in (87):

\[
S ightarrow \text{NP} \quad \text{ADVP} \quad \text{VP} \\
(\uparrow \text{SUBJ}) = \downarrow \quad (\uparrow \text{SADJ}) = \downarrow \quad \uparrow = \downarrow \\
(\tau \uparrow \text{SADJ XCOMP}) = \tau \uparrow
\]

\textit{just}: ADV, (\uparrow \text{PRED}) = just
\textit{fall}: V, (\uparrow \text{PRED}) = fall
\textit{\textit{tomber}}: V, (\uparrow \text{PRED}) = \textit{tomber}

Here the $\tau$ annotation to ADVP states that the $\tau$ of the mother f-structure is the XCOMP of the $\tau$ of the SADJ slot. This set of equations (along with others of a more trivial nature) produces the f-structure (88):

\[
\begin{align*}
\text{SUBJ} & \quad \text{PRED} \quad \text{‘baby’} \\
\text{TENSE} & \quad \text{PAST} \\
\text{SPEC} & \quad \text{the} \\
\text{PRED} & \quad \text{‘fall([baby])’} \\
\text{SADJ} & \quad \{ \text{PRED} \quad \text{‘just’} \}
\end{align*}
\]

However, Sadler et al. (1989, 1990) show that neither approach is able to deal elegantly and straightforwardly with more complex cases of headswitching, as in (89):
(89) I think that the baby just fell $\leftrightarrow$ Je pense que le bébé vient de tomber.

In (89), the headswitching phenomenon takes place in the sentential COMP, rather than in the main clause, as in (83). Here the structure in (85) must be a COMP to a PRED in a higher f-structure. Hence, the normal f-description on embedded S nodes ($\uparrow$ COMP = $\downarrow$) must be optional, and instead the structure in (85) must be unified to the root f-structure as the value of its COMP node. This can be handled by the disjunction in (90):

$$
\text{VP} \rightarrow \text{V that} \quad \text{S} = \{ (\uparrow \text{COMP}) = \downarrow, (\uparrow \text{COMP ARG}) = \downarrow \}
$$

We require this disjunction on embedded S nodes to include $(\uparrow \text{COMP ARG}) = \downarrow$ just in case they contain such a headswitching construction, as f-structure (91) shows:

$$
\begin{align*}
\text{SUBJ} & \quad \begin{array}{|c|c|}
\hline
\text{PRED} & \text{‘I’} \\
\hline
\end{array} \\
\text{PRED} & \text{‘think([held],[just])]’} \\
\text{COMP} & \quad \begin{array}{|c|c|}
\hline
\text{PRED} & \text{‘just([fall])’} \\
\text{SUBJ} & \begin{array}{|c|c|}
\hline
\text{SPEC} & \text{‘baby’} \\
\hline
\end{array} \\
\text{TENSE} & \text{PAST} \\
\text{PRED} & \text{‘fall([baby])’} \\
\hline
\end{array}
\end{align*}
$$

Otherwise, the structure (85) (rooted in just) is not connected to the higher COMP slot. Nevertheless, the solution proposed in (90) seems a little ad hoc, requiring a disjunction just in case the sentential COMP includes a headswitching case. We shall see in the next section that if such headswitching adverbs co-occur, then further disjuncts are required, unless these can be abbreviated by a functional uncertainty equation (cf. (107)).

If we choose the second approach (87), where ‘just’ is a sentential modifier, given that the headswitching is a $\tau$-operation, we require the lexical entry for think in (92):

$$
\text{think: V, } (\uparrow \text{PRED}) = \text{‘think(}[\uparrow \text{SUBJ}(\uparrow \text{COMP})])’
$$

$(\tau \uparrow \text{PRED FN}) = \text{penser}$

$(\tau \uparrow \text{SUBJ}) = (\tau \uparrow \text{SUBJ})$

$(\tau \uparrow \text{COMP}) = (\tau \uparrow \text{COMP})$

This specifies that $\tau$ of the mother f-structure’s COMP slot is the COMP of the $\tau$ of the mother’s f-structure. That is, both this argument, the COMP, and the SUBJ of think are to be translated straightforwardly. This is indeed the case in (93):

$$
\text{I think that the baby fell $\leftrightarrow$ Je pense que le bébé est tombé.}
$$

However, when the COMP includes a headswitching case, as in (89), we end up with a doubly rooted target f-structure because of a clash between the regular $\tau$-equation in the lexical entry for think, (92), and the
structural ν-equation on the ADVP in the (87), which requires the ν of the same piece of f-structure to be the XCOMP of the ν of the SADJ slot. One piece of f-structure is required to fill two inconsistent slots. We will now illustrate this in detail.

The c- and f-structures for the source sentence in (89), I think that the baby just fell, are shown in (94):

(94)

\[
\begin{array}{c}
S_{f_1} \\
\text{NP}_{f_2} \quad \text{VP}_{f_3} \\
I \quad \text{V} \quad \text{that} \\
\text{think} \quad \text{NP}_{f_5} \quad \text{ADVP}_{f_6} \quad \text{VP}_{f_7} \\
\text{the baby} \quad \text{just} \quad \text{fell} \\
\end{array}
\]

\[
\begin{array}{c}
\text{SUBJ} \quad f_2 \{ \text{PRED} \quad \text{?'think'} \} \\
\text{PRED} \quad \text{?'think([i],[fall])'} \\
\text{COMP} \quad f_4, f_7 \\
\text{TENSE} \quad \text{PAST} \\
\text{PRED} \quad \text{?'fall([baby])'} \\
\text{SADJ} \quad \{ f_6 \{ \text{PRED} \quad \text{?'just'} \} \}
\end{array}
\]

The set of τ-equations obtained are those in (95):
(95) From the lexical entry for *think*, (92):
\[ \tau_{f_3} \text{ PRED} = \text{penser} \]
\[ \tau_{f_3} \text{ SUBJ} = ((\tau_{f_3}) \text{ SUBJ}) \]
\[ \tau_{f_3} \text{ COMP} = ((\tau_{f_3}) \text{ COMP}) \]

From the rules and entries in (87):
\[ \tau_{f_6} \text{ PRED} = \text{venir} \]
\[ \tau_{f_7} \text{ PRED} = \text{tomber} \]
\[ \tau_f = (\tau_f \text{ SADJ}) \text{ XCOMP} \]

From straightforward equations for the NPs:
\[ \tau_{f_2} \text{ PRED} = \text{je} \]
\[ \tau_{f_5} \text{ PRED} = \text{bébé} \]
\[ \tau_{f_5} \text{ SPEC} = \text{le} \]

Now, from the c-structure in (94), we see that \( f_3 \)'s COMP is \( f_4 \), so the equation \( \tau_{f_3} \text{ COMP} = ((\tau_{f_3}) \text{ COMP}) \) can be altered quite simply to \( \tau_{f_4} = ((\tau_{f_3}) \text{ COMP}) \). By the same token, \( f_4 \)'s SADJ is \( f_6 \). We now have two equations which cannot both be solved with the result that a target f-structure is formed. The clash is shown in (96):

(96) \[
\begin{align*}
\tau \uparrow \text{ COMP} &= (\tau \uparrow \text{ COMP}) \quad \tau_{f_4} = ((\tau_{f_3}) \text{ COMP}) \\
\tau \uparrow &= \tau \uparrow \text{ SADJ XCOMP} \quad \tau_{f_4} = ((\tau_{f_6}) \text{ XCOMP})
\end{align*}
\]

This results in the construction of the two partial target f-structures in (97):

(97) \[
\begin{align*}
\tau_{f_6} &\begin{cases}
\text{SUBJ} & \tau_{f_5} \\
\text{PRED} & \text{venir([bébé],[tomber])} \\
\text{XCOMP} & \tau_f \begin{cases}
\text{SUBJ} & \tau_{f_6} \begin{cases}
\text{PRED} & \text{penser([je],[…])} \\
\text{SPEC} & \text{le} \\
\text{PRED} & \text{tomber([bébé])}
\end{cases}
\end{cases}
\end{cases}
\end{align*}
\]

\[
\begin{align*}
\tau_{f_3} &\begin{cases}
\text{SUBJ} & \tau_{f_2} \begin{cases}
\text{PRED} & \text{Je'}
\end{cases}
\end{cases}
\end{align*}
\]

\[
\begin{align*}
\tau_{f_4} &\begin{cases}
\text{PRED} & \text{penser([je],[…])} \\
\text{COMP} & \text{le}
\end{cases}
\end{align*}
\]

That is, \( \tau_{f_4} \) is required to be both the XCOMP of *venir* and the COMP of *penser* simultaneously, a conflict that needs to be resolved if a proper target f-structure is to be produced. As was done in (90), we might disjoin the problematic \( \tau \) equation in (92) with an equation for the ‘special’ case in (98):

(98) \[
\tau \uparrow \text{ COMP} = \tau \uparrow \text{ COMP SADJ}
\]

That is, \( \tau \) of the mother at the COMP slot is \( \tau \) of the mother’s COMP SADJ slot. This is clearly undesirable, since it must be specified for every embedding verb.
3.2.2 Scoping of Multiple Adverbs

In a similar fashion, Sadler et al. (1989, 1990) show that the two approaches of Kaplan et al. cannot deal straightforwardly with other complex cases of headswitching involving scoping of multiple adverbs, as in (99):

(99) a. Jan zweent toeavlig graag ----> 
    b. John happens to like to swim 
    c. *John likes to happen to swim 

That is, despite the fact that (99c) is a grammatical string of English, it is not the translation of (99a)---(99b) is. (99c) is the translation of Jan zweent graag toeavlig. Given that the word order differs, we have a different scoping of the adverbs, resulting in a different translation. Let us assume the S rule in (87), and the lexical entries in (100):

(100) graag: ADV, (↑ PRED) = graag  
(τ↑ PRED FN) = like  
zwemmen: V, (↑ PRED) = zwemmen  
(τ↑ SUBJ) = τ(↑ SUBJ)  
(τ↑ PRED FN) = swim

Let us also assume the somewhat simpler sentence (101):

(101) Jan zweent graag (----> John likes to swim) 

Given the S rule in (87), and the lexical entries in (100), the f-structure for (101) is (102):

(102) 

This object is very similar to the f-structure in (88). The other possibility is that the lexical entry for graag is (103):

(103) graag: 
(↑ PRED ) = ‘graag((↑ ARG))’ 
(τ↑ PRED FN) = like 
(τ↑ XCOMP) = τ(↑ ARG) 

In this case, the f-structure in (104) would be built for the Dutch sentence in (101):
(104) \[
\begin{array}{c}
PRED \quad \text{‘graag([zwemen])’} \\
\text{SUBJ} \quad \text{‘Jan’} \\
\text{ARG} \quad \text{TENSE} \quad \text{PRES} \\
PRED \quad \text{‘zwemen([Jan])’}
\end{array}
\]

Being an adverb of the same type, toevallig can occur in similar contexts as graag, as in (105):

(105) Jan zwemt toevallig (→ John happens to swim)

As seen in (99), such adverbs can co-occur. To maintain this approach and produce the translation (99b), we require the f-structure in (106):

(106) \[
\begin{array}{c}
PRED \quad \text{‘toevallig([graag])’} \\
\text{PRED} \quad \text{‘graag([zwemen])’} \\
\text{ARG} \quad \text{TENSE} \quad \text{PRES} \\
PRED \quad \text{‘zwemen([Jan])’}
\end{array}
\]

To associate the embedded S node with an f-structure which is the value of COMP ARG ARG of the mother S’s associated f-structure, we would need to add \((\uparrow \text{COMP ARG ARG}) = \downarrow\) to the disjunction on S in (90). We may be able to simplify these equations using functional uncertainty (Kaplan & Zaenen, 1995), as in (107):

(107) \((\uparrow \text{COMP ARG}*) = \downarrow\)

Given the patterns identified here, there are an infinite number of potential equations available which are impossible to enumerate in a finite disjunction on a grammar rule. Using this extension to the original theory of LFG, uncertainty in grammatical equations can be represented as a regular expression over function names by using the Kleene star operator. Despite this extension, it remains very unnatural to annotate embedded S symbols to allow for the possibility that they may contain (any number of) subcategorizing ADVs of this type. It should be clear that cases like (108) are problematic in the same way as (89)-(98):

(108) Ik denk dat Jan toevallig graag zwemt.

The alternative f-structure corresponding to (99a) is (109):

80
Given the original formulation of LFG, there is no way of producing the required embedding of gråag (‘likewise’) under toevalilig (‘by chance’), and not vice versa, without resorting to tuning: under both approaches outlined, changing our c-structure assumptions to deal with a difficult translation case necessitates the abandonment of modularity.

In general, this approach requires the tuning of f-structures, which is arguably as problematic as producing a sufficiently abstract representation for simple transfer in other systems. For example, since German and Dutch both have such adverbs, as shown in (110), it is possible to treat them as ‘normal’ adverbs and still produce adequate translations:

(110) a. NL: toevalilig —> DE: zufällig  
     b. NL: gråag —> DE: gerne

Hence the danger exists of producing different source language f-structures according to the target language requirements: the cases where the adverbs are top-level PREDs (such as (103), for instance) is appropriate for translation from Dutch to English, just because this involves the switching of heads. The alternative lexical form for the same adverb, (100), is required for translation from Dutch to German, where there is no headswitching for this example.

**Default vs. Specific Translations**

Satisfying the requirement that only possible translations are produced is problematic where the translation of a lexical head is conditioned in some way by its dependants, as in (111):

(111) commit suicide —> se suicider

The problem is that in these cases, suppressing the wrong, compositional translation in LFG-MT is impossible. For instance, we require the default rules in (112):

(112) a. commit —> commettre
     b. suicide —> suicide

Such rules are expressed in LFG-MT by the lexical entries in (113):
\[(113) \quad \text{commit:} \]
\[
(\tau \uparrow \text{PRED } ) = \text{commettre} \\
(\tau \uparrow \text{SUBJ}) = (\tau \uparrow \text{SUBJ}) \\
(\tau \uparrow \text{OBJ}) = (\tau \uparrow \text{OBJ})
\]

\[
\text{suicide:} \\
(\tau \uparrow \text{PRED } ) = \text{suicide}
\]

These entries show how \textit{commit} and \textit{suicide} are to be translated under normal circumstances, such as in (114):

\[(114) \quad \begin{array}{l}
\text{a. Jean commet un crime} \leftrightarrow \text{John commits a crime} \\
\text{b. Le suicide est tragique} \leftrightarrow \text{Suicide is tragic}
\end{array}
\]

Nevertheless, given the default, compositional entries in (113), LFG-MT produces the wrong translation in (115):\footnote{Note that the rules in (112) are \textit{bona fide} translation rules that any English-French MT system will require. It is, therefore, the task of the French generation component to explicitly rule out the incorrect translation in (115), \textit{not the} transfer component. One such possibility would be the use of ‘blocking’ (Poser, 1992), where the existence of an irregular form prevents the corresponding regular form from being used, which otherwise would be expected to occur. Poser shows that despite most blocking being restricted to the lexicon, certain cases of blocking of phrasal constructions by lexical items also seem to occur.}

\[(115) \quad \text{John commits suicide} \leftrightarrow ^* \text{Jean commet le suicide}
\]

LFG-MT \textit{can}, however, derive the correct translation \textit{John se suicide} in such cases via the solution in (116):

\[(116) \quad \text{commit:} \]
\[
(\tau \uparrow \text{PRED FN} ) = \text{se suicider} \\
(\tau \uparrow \text{SUBJ}) = (\tau \uparrow \text{SUBJ}) \\
(\tau \uparrow \text{OBJ PRED}) = \text{c suicide}
\]

Here the collocaotional units ‘\textit{commit} + \textit{suicide}’ are linked as a whole to \textit{se suicider}. The =c equation is a constraining equation: rather than expressing mere equality, it constrains the PRED value of the OBJ of \textit{commit to suicide} when it is to be translated as a whole into \textit{se suicider}. The selective use of constraining equations enables correct translations to be derived which would only be possible in other systems by tuning. Nevertheless, the point remains that in LFG-MT we would get both translations here, i.e. a correct one and a wrong one, since it is not possible to enforce the requirement that specific rules ought to override the default translational rules where applicable, We shall see in chapter 5 that while monolingual DOP language models may still prefer the default translation to the specific alternative, DOT (and LFG-DOT, in chapter 6) bilingual treebanks may prefer the specific translation over that derived using the default rules.

It is appropriate to comment here on the content of lexical entries in LFG-MT. One can see from (116), for example, that the LFG-MT approach of Kaplan \textit{et al.} (1989) does not distinguish between separate source, target and transfer lexica. Other approaches (Sadler \textit{et al.} 1989, 1990) recommend the splitting of such
resources along traditional lines. For example, the $=_{c}$ equation, in conjunction with the first $\tau$-equation in (116), is intended to be a translational constraint. Nevertheless, given that these two equations are not inextricably linked, the $=_{c}$ equation may affect monolingual processing in an undesirable way. That is, if there is no other entry for commit, such as that in (113), then if (116) is used to process English strings with no regard for their translation, all object NPs of commit other than suicide will presumably be ruled out. We discuss the composition of LFG lexica further in section 3.3.1.

Translation using Restriction

Given cases with adverbial modifiers such as (99), p.79, Kaplan & Wedekind (1993) attempt to solve them by the introduction of the notion of restriction, which seeks to overcome problems in mapping between flat syntactic f-structures to hierarchical semantic ones. The intuition is that in such cases semantic units correspond to subsets of functional information, and restricting the f-structure (in other words, removing graag and toeavilig in turn from the adjunct set in (109)) enables (109) to be associated with the alternative s-structures in (117):

\[
\begin{align*}
(117) \text{a.} \quad & \begin{bmatrix}
\text{REL} & \text{'toeavilig'} \\
\text{ARG1} & \begin{bmatrix}
\text{REL} & \text{'graag'} \\
\text{ARG1} & \begin{bmatrix}
\text{REL} & \text{'zwemmen'} \\
\text{ARG1} & \text{'Jan'}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix} \\
\text{b.} \quad & \begin{bmatrix}
\text{REL} & \text{'graag'} \\
\text{ARG1} & \begin{bmatrix}
\text{REL} & \text{'toeavilig'} \\
\text{ARG1} & \begin{bmatrix}
\text{REL} & \text{'zwemmen'} \\
\text{ARG1} & \text{'Jan'}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
\end{align*}
\]

Kaplan & Wedekind (op cit., p.199) define the restriction of an f-structure by an element of an element’s set-value, as in (118):

\[
(118) \quad \text{If } f \text{ is an f-structure and } a \text{ is an attribute:} \\
\quad f \setminus \langle a, g \rangle = \\
\quad \begin{cases} 
\quad f \setminus a & \text{if } (f - a) - \{g\} = \emptyset \\
\quad f \setminus a \cup \{\langle a, (f - a) - \{g\}\rangle\} & \text{otherwise}
\end{cases}
\]

That is, the restriction of an f-structure $f$ by a particular member of an attribute $a$’s set-value is the f-structure which results from deleting that member of the set, and also the attribute itself if an empty set results. We can illustrate how restriction works in (119), taking (109) as input:
We are now in a position to describe the semantic correspon- dences for sentences containing adverbs, such as (99a), p.79, using the restriction operator. Let \( f \) be the \( f \)-structure \((109)\), \( g \) the \( f \)-structure corresponding to \textit{graag} and \( t \) the \( f \)-structure corresponding to \textit{toeavilig}. We give in (120) the constraints necessary to map \((109)\) into the \( s \)-structure \((117b)\), where \textit{graag} has wide scope:

\[(120)\]  
\[\text{a. } (\sigma f \text{ REL}) = (\sigma g \text{ REL}) \]
\[\text{b. } (\sigma f \text{ ARG1}) = \sigma[f \setminus \langle \text{ADJUNCT } g \rangle] \]
\[\text{c. } (\sigma[f \setminus \langle \text{ADJUNCT } g \rangle] \text{ REL}) = (\sigma t \text{ REL}) \]
\[\text{d. } (\sigma[f \setminus \langle \text{ADJUNCT } g \rangle] \text{ ARG1}) \]
\[= \sigma[f \setminus \langle \text{ADJUNCT } g \ t \rangle] \]
\[= \sigma[f \setminus \text{ADJUNCT}] \]

(120a,b) describe the outermost REL and ARG1 configuration in \((117b)\), and (120c,d) describe the next level of embedding at \( s \)-structure. These constraints allow \( f \)-structure subsumption relations to be mapped into the desired hierarchical \( s \)-structures. However, we note that the number of such constraints will grow in proportion with the size of the set of adjuncts. Kaplan & Wedekind (\textit{op cit.}, p.200) give a rule which generates codescription constraints, as in (121):

\[(121)\]  
\[\text{For } f \text{ an } f \text{-structure, } g \in (f \text{ ADJUNCT}), \text{ and } g \text{ a sentence adverb,} \]
\[\sigma f = \sigma g, \text{ and} \]
\[\sigma[f \text{ ARG1}] = \sigma[f \setminus \langle \text{ADJUNCT } g \rangle] \]

(121) allows each element to be selected non-deterministically from an adjunct set to contribute to the relation for the \( s \)-structure of the enclosing \( f \)-structure. Furthermore, the \( s \)-structure corresponding to the \( f \)-structure minus the selected member becomes the ARG1 of that relation. Given that the restriction operation applies non-deterministically to all members of a set, we get, as here, a correct and some incorrect structures: with respect to sentence \((99)\), p.79, \( s \)-structure \((117a)\) is correct whilst \((117b)\) is incorrect. Although this may be seen as an improvement on the codescription approach, where the production of the correct \( f \)-structure could not be guaranteed, it nevertheless leaves something to be desired in that human intervention is necessary to manually select the optimal structure from the (possibly large) set of candidate solutions.
This addition to LFG-MT, while partially solving some of the immediate problems, has met with other criticisms, notably those of Butt (1994), who states that the use of the restriction operator entails dealing with complex predicates in Urdu in an unintuitive way. Butt consequently advocates the use of linear logic source and target semantic representations (Dalrymple et al., 1993) as a more flexible solution to this problem, in conjunction with the classical $\tau$-equations and language specific (i.e. monolingual) mapping principles between f- and s-structures. The use of linear logic has also recently been adopted by Van Genabith et al. (1998), who recommend it as a formalism for the representation of transfer, in addition to performing transfer on linear logic meaning constructors. We describe this approach further in the following section.

### 3.2.3 NPs containing Modifiers

Some NPs containing modifiers lead to headswitching in translation. Such modifiers include (122):

(122) a hundred N $\Rightarrow$ une centaine de N  
     a draft N $\Rightarrow$ un projet de N

Again, given the original formulation of LFG-MT (Kaplan et al., 1989), there are two possible ways in which we can attempt to handle these. Let us first deal with the translation in (123):

(123) He saw an attempted murder $\implies$ Il a vu une tentative de meurtre.

Let us assume the structural rules in (124):

```
S $\rightarrow$ NP         VP
   (↑ SUBJ)=↓    ↑=↓

VP $\rightarrow$ V         NP
   ↑=↓          (↑OBJ)=↓

(124)

NP $\rightarrow$ (Det)     AP*         N
   (↑SPEC)=↓    ↓ε (↑ADJ)   ↑=↓
   τ ↓ OBL OBJ = τ ↑

AP $\rightarrow$ Adj
   ↑=↓
```

Given these rules, the c-structure tree in (125) can be formed:
Solving the f-description for this sentence produces the (simplified) English f-structure in (126):

```
(126)  \( f_0, f_2 \)
       \[
         \begin{align*}
           \text{SUBJ} & f_1 \left[ \begin{array}{c}
             \text{PRED} \\
             \text{'he'}
           \end{array} \right] \\
           \text{PRED} & \text{see}((\uparrow \text{SUBJ})(\uparrow \text{OBJ})) \\
           \text{TENSE} & \text{PAST} \\
           \text{OBJ} & f_3 \left[ \begin{array}{c}
             \text{SPEC} \\
             \text{'an'}
           \end{array} \right] \\
           \text{ADJUNCT} & \{ f_4 \left[ \begin{array}{c}
             \text{PRED} \\
             \text{'attempted'}
           \end{array} \right] \}
         \end{align*}
       \]
```

In order to try and achieve the translation in (123), the lexical entries in (127) are required:

```
(127)  \text{see: } V, (\tau \uparrow \text{PRED FN}) = \text{voir} \\
       (\tau \uparrow \text{SUBJ}) = \tau (\uparrow \text{SUBJ}) \\
       (\tau \uparrow \text{OBJ}) = \tau (\uparrow \text{OBJ})
```

\textit{attempted:}

```
(\tau \uparrow \text{PRED FN}) = \text{tentative} \\
(\tau \uparrow \text{OBL PRED}) = \text{de}
```

\textit{he:}

```
(\tau \uparrow \text{PRED FN}) = \text{il}
```

\textit{murder:}

```
(\tau \uparrow \text{PRED FN}) = \text{meurtre}
```

Translating \( f_3 \) on its own, we have the lexical equations from \textit{attempted} and \textit{murder} in (127), and the structural translation equations in (124). With metavariable substitution, the descriptions in (128) are derived:
(128)  
murder:
\[ \tau f_3 \text{ PRED FN} = \text{meurtre} \]

 attempted:
\[ \tau f_4 \text{ PRED FN} = \text{tentative} \]
\[ \tau f_4 \text{ OBL PRED} = \text{de} \]

From the equation on the AP node in (124):
\[ \tau f_4 \text{ OBL OBJ} = \tau f_3 \]

Solving these equations enables the f-structure in (129) to be derived:

\[
(129) \quad \tau f_4 = \begin{bmatrix}
\text{PRED} & \text{tentative} \\
\text{OBL} & \begin{bmatrix}
\text{PRED} & \text{de} \\
\text{OBJ} & \tau f_3 \end{bmatrix}
\end{bmatrix}
\]

If we now produce the rest of the \( \tau \)-equations, we have (130):

\[
(130) \quad \text{see:} \\
\tau f_0 \text{ PRED FN} = \text{voir} \\
\tau f_0 \text{ SUBJ} = \tau (f_0 \text{ SUBJ}) = \tau f_1 \\
\tau f_0 \text{ OBJ} = \tau (f_0 \text{ OBJ}) = \tau f_3
\]

 he:
\[ \tau f_1 \text{ PRED FN} = \text{il} \]

Now we see that the equations from lexical and structural rules cause equation clashes. In attempting to produce the translation in (123), we obtain the equation pair in (131):

\[
(131) \quad \text{a. } \tau f_3 = \tau f_4 \text{ OBL OBJ (from the NP rule in (124))} \\
\text{b. } \tau f_3 = \tau f_0 \text{ OBJ (from (127), the lexical entry for see)}
\]

What needs to be done is to insert the French f-structure (129) into the OBJ slot of the sentential f-structure \( \tau f_0 \). The only way seems to be to annotate see\(^2\) in an \textit{ad hoc} way, just in case its object has to be headswitched, as in (132):

\[
(132) \quad \text{see:} \\
\tau (\uparrow \text{PRED FN}) = \text{voir} \\
\tau (\uparrow \text{SUBJ}) = (\tau \uparrow \text{SUBJ}) \\
\{ \tau (\uparrow \text{OBJ}) = (\tau \uparrow \text{OBJ}), \tau (\uparrow \text{OBJ OBL OBJ}) = (\tau \uparrow \text{OBJ}) \}
\]

Using the second of these disjunctive equations ‘undoes’ the clash, as (133) illustrates:

\[\textit{And every other verb of its type. Alternatively, this might be coped with by means of a lexical redundancy rule, on the assumption that such rules used freely in monolingual LFG analysis are permitted in LFG-MT.}\]

\[ \text{87} \]
(133)  a.  \( \tau f_4 \) OBL OBJ = \( \tau f_3 \) (from the NP rule in (124))
    
    b.  \( \tau f_0 \) OBL OBJ OBJ = \( \tau f_3 \) (from (132), the revised lexical entry for \textit{see})

Now the correct f-structure (134) can be derived:

\[
\begin{array}{c}
\text{SUBJ} \quad f_1 \left[ \begin{array}{c}
\text{PRED} \quad \text{‘il’}
\end{array} \right] \\
\text{PRED} \quad \text{‘voir}(\uparrow\text{SUBJ})(\uparrow\text{OBJ})’
\end{array}
\]

\[
\begin{array}{c}
\text{TENSE} \quad \text{PAST}
\end{array}
\]

\[
\begin{array}{c}
\text{OBJ} \quad f_2 \\
\text{OBL} \quad f_3 \\
\text{OBJ} \quad f_4 \\
\text{OBJ} \quad f_5 \left[ \begin{array}{c}
\text{PRED} \quad \text{‘de’}
\end{array} \right] \\
\text{OBJ} \quad f_6 \left[ \begin{array}{c}
\text{PRED} \quad \text{‘meurtre’}
\end{array} \right]
\end{array}
\]

This extra annotation would have to be added to any slot that might contain an NP containing a modifier that causes headswitching in translation, like those in (122).\(^3\) However, the presence of two headswitching modifiers in the same NP causes additional problems, so that when faced with sentence (135), we are forced to add further disjunctions into the equation set for \textit{see} (and again, for all other verbs of the same type):

(135)  He saw a hundred attempted murders.

This is illustrated in (136):

(136)  \textit{see}:

\[
\{ \tau \uparrow \text{OBJ} = \tau ( \uparrow \text{OBJ} ), \quad \\
\tau \uparrow \text{OBJ OBL OBJ} = \tau ( \uparrow \text{OBJ} ), \quad \\
\tau \uparrow \text{OBJ OBL OBJ OBL OBJ} = \tau ( \uparrow \text{OBJ} ) \}
\]

This disjunction is required in order to produce the (simplified) target f-structure in (137):

\[
\begin{array}{c}
\text{SUBJ} \quad f_1 \left[ \begin{array}{c}
\text{PRED} \quad \text{‘il’}
\end{array} \right] \\
\text{OBL} \quad f_3 \\
\text{OBJ} \quad f_4 \\
\text{OBJ} \quad f_5 \left[ \begin{array}{c}
\text{PRED} \quad \text{‘de’}
\end{array} \right] \\
\text{OBJ} \quad f_6 \left[ \begin{array}{c}
\text{PRED} \quad \text{‘meurtre’}
\end{array} \right]
\end{array}
\]

More would need to be added if we could find examples of three such NPs, and so on. Therefore, this methodology must be rejected as \textit{ad hoc}.

The alternative approach of Kaplan \textit{et al.} leads to predicates like \textit{attempted} and \textit{hundred} having their head noun as a dependant in English f-structure, as in (138):

\[^3\text{We note that with the emergence of ‘inside-out’ functional uncertainty (Dairyple, 1993:117ff.), it might be preferable nowadays to lexicalize these ‘special’ equations, rather than having them on the rules.}\]
This approach causes headswitching within English analysis. That is, the more natural analysis of such NPs is with the noun as head, not the adjectival adjunct (as in (126)), but this approach forces us to consider this alternative possibility. This leads to an interesting situation as *hundred* is translationally ambiguous, as (139) shows:

(139) a. *hundred* $\rightarrow$ *centaine*, in which case the translation is headswitching,

b. *hundred* $\rightarrow$ *cent*, in which case it is not.

Thus we have to have two English f-structures. The first contains *hundred* as a governor of its noun, to prepare for *centaine*, as in (140):

(140) $\begin{bmatrix}
\text{PRED} & \text{‘hundred’} \\
\text{SPEC} & \text{‘a’} \\
\text{ARG} & \begin{bmatrix}
\text{PRED} & \text{‘attempted’} \\
\text{ARG} & \begin{bmatrix}
\text{PRED} & \text{‘murders’} \\
\text{SPEC} & \text{‘a’} \\
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}$

The second f-structure contains *hundred* as a modifier of its noun, to prepare for *cent*, as in (141):

(141) $\begin{bmatrix}
\text{PRED} & \text{‘attempted’} \\
\text{ARG} & \begin{bmatrix}
\text{PRED} & \text{‘murders’} \\
\text{ARG} & \begin{bmatrix}
\text{PRED} & \text{‘hundred’} \\
\text{SPEC} & \text{‘a’} \\
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}$

Thus an English ambiguity is postulated for no other reason than to prepare for a translational one. The dangers of tuning are particularly obvious in such a case. Moreover, the monolingual solution suffers from the same ad hoc-ness as the bilingual. Let us assume the f-structure (142):

(142) $\begin{bmatrix}
\text{f}_0, f_2 & \text{PRED} & \text{see} \\
\text{SUBJ} & f_1 & \begin{bmatrix}
\text{PRED} & \text{he} \\
\text{OBJ} & f_4 & \begin{bmatrix}
\text{PRED} & \text{attempted} \\
\text{ARG} & f_3 & \begin{bmatrix}
\text{PRED} & \text{murder} \\
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}$

To obtain the pairing of c-structure (125) with f-structure (142), one needs the equation in (143) on the AP slot in the c-structure rule for NP:

(143) $\begin{bmatrix}
\text{NP} & \rightarrow & (\text{Det}) & \text{AP*} & \text{N} \\
(\uparrow \text{SPEC}) = \downarrow & \uparrow = (\downarrow \text{ARG}) & \uparrow = \downarrow
\end{bmatrix}$

This enables us to obtain $f_3 = f_4$ ARG. Furthermore, we require the equation (144) on the c-structure rule for VP to obtain eventually $f_0$ OBJ ARG = $f_3$:
Let us assume the c-structure in (145):

\[
S, f_0 \\
\downarrow
\]
\[
\text{NP, } f_1 \quad \text{VP, } f_2 \\
\downarrow \quad \downarrow
\]
\[
\text{he} \quad \text{saw} \quad \text{Det} \quad \text{AP, } f_4 \quad \text{NP, } f_3 \\
\downarrow \quad \downarrow \quad \downarrow \quad \downarrow
\]
\[
\text{a} \quad \text{Adj} \quad \text{AP, } f_5 \quad \text{N} \\
\downarrow \quad \downarrow
\]
\[
\text{hundred} \quad \text{Adj} \quad \text{murders} \quad \text{attempted}
\]

Consider also the f-structure in (146):

\[
f_0, f_2 \\
\downarrow
\]
\[
\text{PRED see} \quad \text{SUBJ } f_1 \quad \text{PRED he} \\
\downarrow \quad \downarrow \quad \downarrow
\]
\[
\text{OBJ } f_5 \quad \text{ARG } f_4 \quad \text{ARG } f_3 \quad \text{ARG } f_5 \quad \text{PRED murder}
\]

In order to pair the objects in (145) and (146), we need the possibility (147) on the AP node in (143) to obtain \( f_3 = f_5 \) ARG ARG:

\[
\uparrow = \downarrow \text{ (ARG ARG)}
\]

The equation (148) is also required on the object NP node in (144) to obtain eventually \( f_0 \) OBJ ARG ARG = \( f_3 \):

\[
\uparrow \text{ (OBJ ARG ARG)} = \downarrow
\]

These extensions are obviously closely parallel to those required in the bilingual approach. Again, we presume that these equations can be simplified using functional uncertainty, giving (149) alone on the AP node in (143):

\[
\uparrow = (\downarrow \text{ ARG*})
\]

We also have (150) alone on the object NP node in (144):
(150) \( \owns \text{OBJ ARG*} \) = \( \downarrow \)

Nevertheless, the ad hoc-ness of (150) in particular remains. Presumably approaches which lead to monolingual headswitching should not be countenanced, unless absolutely necessary. It is a given in translation; surely it should be avoidable monolingually.

Kaplan & Wedekind (1993) propose that examples such as (135), p.88, can be solved in a similar way to multiple adjuncts, by the use of the restriction operator, with transfer performed on s-structures rather than on f-structures. However, in this instance we have headswitching in both source and target, as (151) and (152) show:

\[
(151) \begin{align*}
\text{a.} & \quad \begin{array}{c}
\text{REL} \quad \text{‘attempted’} \\
\text{ARG1} \quad \begin{array}{c}
\text{REL} \quad \text{‘murders’} \\
\text{ARG1} \quad \text{‘hundred’}
\end{array}
\end{array} \\
\text{b.} & \quad \begin{array}{c}
\text{PRED} \quad \text{‘hundred’} \\
\text{ARG} \quad \begin{array}{c}
\text{PRED} \quad \text{‘attempted’} \\
\text{ARG} \quad \begin{array}{c}
\text{PRED} \quad \text{‘murders’}
\end{array}
\end{array}
\end{array} \\
& \quad \begin{array}{c}
\text{PRED} \quad \text{‘attempted’} \\
\text{ARG} \quad \begin{array}{c}
\text{PRED} \quad \text{‘murders’} \\
\text{ARG} \quad \begin{array}{c}
\text{PRED} \quad \text{‘hundred’}
\end{array}
\end{array}
\end{array}
\end{align*}
\]

Both f-structures in (151b) have (151a) as their s-structure, except that there is headswitching between (151b-1) and (151a).

\[
(152) \begin{align*}
\text{a.} & \quad \begin{array}{c}
\text{REL} \quad \text{‘tentatives’} \\
\text{ARG1} \quad \begin{array}{c}
\text{REL} \quad \text{‘meurtres’} \\
\text{ARG1} \quad \text{‘cent/centaine’}
\end{array}
\end{array} \\
\text{b.} & \quad \begin{array}{c}
\text{PRED} \quad \text{‘tentatives’} \\
\text{ARG} \quad \begin{array}{c}
\text{PRED} \quad \text{‘meurtres’} \\
\text{ARG} \quad \begin{array}{c}
\text{PRED} \quad \text{‘cent’}
\end{array}
\end{array}
\end{array} \\
& \quad \begin{array}{c}
\text{PRED} \quad \text{‘centaine’} \\
\text{ARG} \quad \begin{array}{c}
\text{PRED} \quad \text{‘tentatives’} \\
\text{ARG} \quad \begin{array}{c}
\text{PRED} \quad \text{‘meurtres’}
\end{array}
\end{array}
\end{array}
\end{align*}
\]

Both f-structures in (152b) have (152a) as their s-structure, except that there is headswitching between (152b-2) and (152a). (151a) and (152a) are translations of each other using \( \tau’ \), as seen by the equations in (153) (cf. Kaplan et al., 1989:276):

91
(153) a. \( \tau'(\sigma \uparrow \text{ARG1}) = (\sigma \uparrow \text{ARG1}) \)

b. \( \tau'(\sigma \uparrow \text{REL}) = (\sigma \uparrow \text{REL}) \)

(151b-1) and (151b-2), and (151b-1) and (152b-1) are straightforward translations of each other using \( \tau \).

Kaplan & Wedekind (1993) introduced the \emph{restriction} operator partly as an attempt to solve the set of translation problems raised by Sadler \emph{et al.} (1989, 1990). The problems the codescription approach of Kaplan \emph{et al.} (1989) encountered when faced with the whole class of headswitching phenomena were:

“... a symptom of a more fundamental error in the syntactic and semantic analyses of the source language. In sentences with adverbial modifiers (such as (83), p.74, and (99), p.79, above), the syntactic head ... is not the same as the semantic head ... Moreover, normal linguistic arguments would assign a flat f-structure to sentences with several adverbs while meaning relations would be represented in a hierarchical semantic structure. Thus, on this view, if translation codescriptions map from the proper hierarchical semantic structures via \( \tau' \) instead of from flat f-structures via \( \tau \),

adverbial head-switching disappears as a special problem for correspondence-based translation.”

(Kaplan & Wedekind, 1993:196)

Despite these attempts to resolve cases of headswitching, a general solution to this translation problem still seems to fall foul of something inherent in the formalism. The introduction of \emph{restriction} seems to necessitate monolingual headswitching (as in (138), p.88), as well as requiring a further level of structure in the translation process. Sadler & Thompson (1991) point out that reforming the translation codescription problems using restriction may simply move the problems from one area to another. Kaplan & Wedekind respond that “this may be so, but any conceptual clarification in such a murky domain must be regarded as an advance, if only because it helps to spotlight the issues that are relevant to a solution” (Kaplan & Wedekind, 1993:196). Despite such clarification, restriction only partially solves the problematic cases, so that one might argue that the use of s-structures for translation may at least be called into question. It can be said with some confidence that it does not provide a general solution to the problem of headswitching.

**Linear Logic**

Dalrymple \emph{et al.} (1995) state that given that the elements in an f-structure are unordered, it is not very natural to combine meanings of f-structure constituents using \( \lambda \)-calculus. Nor is it in fact useful, as an artificial order on the composition of meanings needs to be imposed, given that semantic compositionality needs to be applied in a strict order. Linear logic (Girard, 1987), meanwhile, is able to specify requirements on the composition of the semantic contribution of lexical items without having to rely on specific surface relations to guide composition. It is able, therefore, to relate f-structures with their meanings at the level of s-structure. Linear logic meaning constructors (premises) are obtained from words, and may be interpreted as ‘instructions’ by which the meaning of the lexical entry may be combined with the meanings of its arguments. Dalrymple \emph{et al.} (1995) note that linear logic is the ‘glue’ which ties meaning constructors together.

Some cases of headswitching, notably adjuncts and embedded headswitching phenomena, have been tackled using linear logic to formalize transfer rules (Van Genabith \emph{et al.}, 1998), maintaining the clear distinction
between meaning language and glue language, following Dalrymple et al. (1995). Nevertheless, not all of the cases described here can be solved without further problems raising their head.

To give a simple example, consider the source f-structure and meaning assignments associated with John swims in (154):

\[
\text{(154)} \quad \begin{bmatrix}
\text{SUBJ} \\
\text{PRED}
\end{bmatrix}
\begin{bmatrix}
\text{PRED} & \text{'swim' [↑ \text{SUBJ}]} \end{bmatrix} \begin{bmatrix}
\begin{cases}
\forall X \ [(f_2)_\sigma \sim \text{john}] \\
\forall X \ [(f_1)_\sigma \sim \text{swim} (X)]
\end{cases} \\
\end{bmatrix}
\]

\[
\vdash (f_1)_\sigma \sim \text{swim(john)}
\]

Given (154), the set of transfer constructors in (155) relate this string to its German equivalent Johannes schwimmt:

\[
\text{(155)}
\]

\[
\text{Trans} = \begin{bmatrix}
\forall F [F_\sigma \sim \text{john} \rightarrow F_\sigma \sim \text{johannes}] \\
\forall F [\forall X((F \text{ SUBJ})_\sigma \sim X \rightarrow F_\sigma \sim \text{swim} (X))] \\
\forall X((F \text{ SUBJ})_\sigma \sim X \rightarrow F_\sigma \sim \text{schwimmen(X)})
\end{bmatrix}
\]

Going from source to target, each transfer constructor in \text{Trans} relates a source meaning constructor \(\sigma\) to a corresponding target meaning constructor \(\tau\): \(\sigma \rightarrow \tau\).

Van Genabith et al. note that especially in more complex cases, “transfer constructors are massively redundant and ... there is nothing to guarantee that our deductions terminate in a set of target meaning constructors (rather than in disambiguated target meaning assignments)”. The whole point of their approach is to perform ambiguity-preserving transfer on underspecified representations, so if the target meaning assignments were disambiguated then the raison d’être of the approach would be compromised, despite the results being correct. Keeping transfer constructors completely separate from the set of meaning constructors by using separate connectives ( \(\rightarrow\) for meaning constructors, and \(\rightarrow_{\text{t}}\) for transfer) ensures that transfer terminates in sets of target meaning constructors. As for redundancy, tautological transfer constructors such as \(\sigma \rightarrow_{\text{t}} \sigma\) are dropped, and when source and target meaning constructors have left-common prefixes, as in \((\tau \rightarrow \sigma) \rightarrow_{\text{t}} (\tau \rightarrow \tau)\), these are reducible to \(\sigma \rightarrow_{\text{t}} \tau\).

Van Genabith et al. go on to show that their approach can handle some more complex cases of transfer problem. They illustrate a case of argument switching with Das Photo ist Hans misslungen \(\rightarrow\) Hans a râter la photo (Hans ruined the photo). This requires the (simpler) transfer constructor in (156):

\[
\text{(156)}
\]

\[
\forall F, X, Y \ (F_\sigma \sim \text{misslingen}(X,Y) \rightarrow_{\text{t}} F_\sigma \sim \text{râter}(X,Y))
\]

From (156), together with the set of instantiated meaning constructors and other appropriate transfer constructors, the set of target meaning constructors in (157) can be derived:
(157)\[
\{ (f_3)_\sigma \leadsto \text{photo} \\
(f_3)_\sigma \leadsto \text{hans} \\
\forall X \forall Y [( (f_3)_\sigma \leadsto X \otimes (f_2)_\sigma \leadsto Y) \leadsto (f_1)_\sigma \leadsto \text{misslingen}(X,Y)]
\}
\cup \{ \text{misslingen} \leadsto \text{räter} \} \vdash \{ \quad 1. (f_3)_\sigma \leadsto \text{photo} \\
2. (f_3)_\sigma \leadsto \text{hans} \\
3. \forall X \forall Y [( (f_3)_\sigma \leadsto X \otimes (f_2)_\sigma \leadsto Y) \leadsto (f_1)_\sigma \leadsto \text{räter}(X,Y)]
\}
\]

The third target meaning constructor correctly binds the meanings of X and Y in the monolingual versions shown in (158):

(158)\[
\text{misslingen: } \forall X \forall Y [( (\uparrow \text{OBJ})_\sigma \leadsto X \otimes (\uparrow \text{SUBJ})_\sigma \leadsto Y) \leadsto (f_\sigma \leadsto \text{misslingen}(X,Y)] \\
\text{räter: } \forall X \forall Y [( (\uparrow \text{SUBJ})_\sigma \leadsto X \otimes (\uparrow \text{OBJ})_\sigma \leadsto Y) \leadsto (f_\sigma \leadsto \text{räter}(X,Y)]
\]

This ensures that the appropriate arguments are ‘switched’ in transfer. Van Genabith et al. also note that “the resource sensitivity of linear logic also fares well with another type of lexical transfer, exemplified by the pair commit suicide and its French translation se suicider” (cf. (112)-(115), pp.81-82), and provide the transfer constructor (159) as a solution:

(159)\[
\forall F, X, Y ((F \text{ OBJ})_\sigma \leadsto \text{suicide} \otimes F_\sigma \leadsto \text{commit}(X,Y)) \leadsto F_\sigma \leadsto \text{se sui}cider(X))
\]

Nevertheless, when it comes to headswitching cases the linear logic approach encounters some problems. Van Genabith et al. use the like \(\leftrightarrow\) gerne example in the sentence pair Hans schwimmt gerne \(\leftrightarrow\) Hans likes swimming. Given the appropriate f-structures, the source set of meaning constructors in (160) is derived:

(160)\[
\begin{align*}
\{ (f_2)_\sigma \leadsto \text{hans} \\
\forall X [(f_2)_\sigma \leadsto X \leadsto (f_1)_\sigma \leadsto \text{schwimmen}(X)] \\
\forall P [(f_1)_\sigma \leadsto P \leadsto (f_1)_\sigma \leadsto \text{gerne}(P)]
\end{align*}
\]

The meaning constructor for like is (161):

(161)\[
\text{like: } \forall X, P [( (\uparrow \text{SUBJ})_\sigma \leadsto X \otimes \forall Y (( (\uparrow \text{XCOMP SUBJ})_\sigma \leadsto Y \leadsto ( (\uparrow \text{XCOMP})_\sigma \leadsto P(Y))) \leadsto (f_1)_\sigma \leadsto \text{gerne}(P))] \leadsto (f_1)_\sigma \leadsto \text{gerne}(P)]
\]

This final term, like\((X,P(X))\), indicates that X is the subject of like, and that somewhere in its second argument, X re-enters as the XCOMP SUBJ. The complete set of meaning constructors for Hans likes swimming is given in (162):

94
The transfer constructor \( \text{gerne} \otimes \text{like} \) consumes the entire meaning constructor for \( \text{gerne} \) as there are no left-common prefixes in the meaning constructors, and the meaning constructor for \( \text{like} \) in (163) is derived:

\[
\forall F \left[ \forall P (F \otimes P \Rightarrow F \otimes \text{gerne}(P)) \right] \\
\forall X, P \left[ (F \ \text{SUBJ})_{\sigma} \Rightarrow X \otimes Y \left( (F \ \text{XCOMP} \ \text{SUBJ})_{\sigma} \Rightarrow Y \Rightarrow F_{\sigma} \Rightarrow P(Y) \Rightarrow F_{\sigma} \Rightarrow \text{like}(X, P(X)) \right) \right]
\]

Now the first problem with this approach can be seen. On the right-hand side of the \( \otimes \), we observe that the meaning constructor has rewritten a node \( F \) rather than a node \( F \ \text{XCOMP} \) to match \( P(Y) \). Consequently, with the instantiated source meaning constructors and the transfer constructors the equations in (164) are produced:

\[
\text{Source} \cup \left\{ \begin{array}{l}
\text{schwimmen} \otimes \text{swim} \\
\text{gerne} \otimes \text{like} 
\end{array} \right\} \Rightarrow_{t} \\
\left\{ \\
1. \ (f_{2})_{\sigma} \Rightarrow \text{hans} \\
2. \ \forall X [(f_{2})_{\sigma} \Rightarrow X \Rightarrow (f_{1})_{\sigma} \Rightarrow \text{swim}(X)] \\
3. \ \forall X, P [(f_{2})_{\sigma} \Rightarrow X \otimes Y [(f_{2})_{\sigma} \Rightarrow Y \Rightarrow (f_{1})_{\sigma} \Rightarrow P(Y))] \Rightarrow (f_{1})_{\sigma} \Rightarrow \text{like}(X, P(X))] \\
\right\} \Rightarrow_{t} \\
\ (f_{1})_{\sigma} \Rightarrow \text{like}(\text{hans}, \text{swim}(\text{hans}))
\]

Comparison of the third target meaning constructor with (161) shows that the transfer operation has rewritten a single node \( (f_{1})_{\sigma} \) rather than accessing a complement node \( (f_{1} \ \text{XCOMP})_{\sigma} \) to match against \( P(Y) \). Van Genabith \textit{et al.} then give an example of embedded headswitching involving the \( \text{like} \leftrightarrow \text{gerne} \) case, and unsurprisingly the same fault is uncovered again.

Transfer should deliver exactly the set of meaning constructors as would be obtained by independent analysis of the target string. If it does not, target language generation from underspecified sets of target meaning constructors will not produce the required output, and the overall translation obtained will be wrong. Van Genabith \textit{et al.} propose to rectify the problem by ‘pushing down’ the predicate-argument nucleus of verbs one (or more, as appropriate) levels, via functional uncertainty over XCOMP. Hence the transfer constructor \( \text{schwimmen} \otimes \text{swim} \) gets amended to (165):

\[
95
\]
\[ (165) \quad schwimmen \rightarrow_{t} \text{swim} \]
\[ \forall F, X (F_x \rightsquigarrow schwimmen(X) \rightarrow_{t} (F \ \text{XCOMP}_x) \rightsquigarrow \text{swim}(X)) \]

Furthermore, the transfer constructor \( \text{gerne} \rightarrow_{t} \text{like} \) is also redefined given the knowledge that the predicate-argument structure and the corresponding semantic projector associated with the translation of the proposition in the scope of the source adjunct is ‘pushed down’ via functional uncertainty in (165) on the target side. Given these amendments, the set of target meaning constructors produced is identical to those in (161), as required.

However, there are a number of problems with this ‘solution’. It is clear that any verb plus associated arguments can occur as the complement of \( \text{like} \). Consequently, the amended transfer constructor in (165) will have repercussions for every verbal translation relation. That is, all transfer constructors relating verbs will need to include such an equation, \( \text{just in case} \) it ever occurs as an XCOMP in an infinitival phrase of another verb. Furthermore, the addition of how to translate verbs such as \( schwimmen \leftrightarrow \text{swim} \) in the event of \( \text{swim} \) appearing as the complement of \( \text{like} \) has nothing to do with the translation relation in question at all: the translation of \( schwimmen \) as \( \text{swim} \) is a case of simple transfer. Information about \( \text{swim} \) as an XCOMP has nothing to do with \( \text{swim} \), or \( schwimmen \), but is an artefact of the \( \text{like} \leftrightarrow \text{gerne} \) case. Consequently it should be removed from the context of \( schwimmen \leftrightarrow \text{swim} \) and relocated in its proper place. If the approach is subsequently unable to deal with headswitching examples, then so be it, but at least the basic translation relations are kept intact and untainted by \textit{ad hoc} information which does not belong there.

\section*{Underspecified Semantic Representations}

Any advantage of encoding transfer relations in terms of semantic representations may be undone by the likelihood that semantic ambiguities may lead to a huge overload in the disambiguation task. There are accordingly a number of ways in which the problem of semantic ambiguity in transfer can be addressed. The approach of (Van Genabith \textit{et al.}, 1998) in the previous section uses packed representations and performs transfer on meaning constructors. A related approach is the use of underspecified semantic representations (Dorna \textit{et al.}, 1998) to try to overcome the headswitching problems presented here.

Dorna \textit{et al.} port the \textit{Verbmobil} transfer component (Dorna & Emde, 1996) to f-structures and incorporate the underspecified semantic interpretation methodology of (Van Genabith & Crouch, 1997) to interface source and target f-structures. As an example, the f-structure and the term representation equivalent for the string \( \text{Hans Kocht gerne} \) are given in (166):

\[
\begin{array}{c}
\text{(166)} \\
\hspace{1cm} \begin{array}{c}
\text{SUBJ} \\
\text{PRED} \\
\text{ADJS}
\end{array}
\begin{array}{c}
\begin{array}{c}
\text{[PRED} \quad \text{‘Hans’}]_{2}
\end{array}
\begin{array}{c}
\begin{array}{c}
\text{[PRED} \quad \text{‘kochen’}]_{3}
\end{array}
\end{array}
\end{array}
\end{array}
\end{array}
\]

\[
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\{ \text{kochen(n1), SUBJ(n1,n2), Hans(n2), ADJS(n1,n3), gerne(n3) } \}
\end{array}
\end{array}
\end{array}
\]

This set of source terms are related to their target equivalents using the transfer rules in (167):

96
(167) a. \{ kochen(E) \} ī\{ cook(E) \}.

b. \{ SUBJ(E,X) \} ī\{ SUBJ(E,X) \}.

c. \{ Hans(X) \} ī\{ Hans(X) \}.

d. \{ ADJS(E,X), gerne(X) \} ī\{ SUBJ(E,Y) \} ī\{ like(X),XCOMP(X,E),SUBJ(X,Y) \}.

The headswitching rule (167d) includes a test on a copy of the original input to the right of the #, which binds Y at runtime when employing the rule from left to right. Here it ensures re-entrant structure sharing of the higher level SUBJ in the XCOMP on the English side. In the reverse direction, such target language tests are ignored (as there is no XCOMP on the German side, there is no need for any re-entrant SUBJ). Once the rules in (167) are applied to the terms in (166), the resultant English terms and f-structure for Hans likes cooking in (168) are derived:

\[
\text{(168)} \begin{bmatrix}
\text{SUBJ} & \text{PRED} & \text{like(\{SUBJ\}(\{XCOMP\}))} \\
\text{PRED} & \text{like(\{SUBJ\}(\{XCOMP\}))} & \text{like(n3),} \\
\text{XCOMP} & \text{SUBJ} & \text{like(n2),} \\
\text{SUBJ} & \text{PRED} & \text{like(n3,n2),} \\
\end{bmatrix} \begin{bmatrix}
\text{like(n3),} \\
\text{SUBJ(n3,n2),} \\
\text{Hans(n2),} \\
\text{XCOMP(n3,n1),} \\
\text{cook(n1),} \\
\text{SUBJ(n1,n2)} \\
\end{bmatrix}
\]

Dorna et al. note that the semantic-based transfer component of (Dorna & Emele, 1996) rewrites underspecified semantic representations, specifically underspecified discourse representation structures (UDRS), whereas f-structures contain syntactic information. They claim that f-structures “encode basic predicate-argument relations, and this is essentially semantic information. Consequently it turns out that there are important structural similarities between f-structures and UDRSs so much so that f-structures can be ‘read’ as UDRSs and hence be assigned an underspecified truth-conditional interpretation defined in terms of embeddings of f-structures in the UDRT formalism”. As an example, the headswitching case in (166)–(168) is mapped to the UDRSs in (169):

\[
\text{(169)} \begin{array}{c}
l_{\rightarrow} : \begin{array}{c}
x_2 : \text{Hans}(x_2) \\
x_2 : \text{Hans}(x_2) \\
\end{array} \begin{array}{c}
\uparrow \\
\uparrow \\
\end{array} \\
l_{\leftarrow} : \begin{array}{c}
l_3 : \text{gerne}(x_3) \\
l_3 : \text{like}(x_3) \\
\end{array} \\
\end{array}
\]

Hans kocht gerne

Hans likes cooking

Dorna et al. observe with respect to (169) that the structural mismatch between the two f-structures in (166) and (168) “has disappeared on the level of UDRS representations and transfer is facilitated”. They then show an example of embedded headswitching with similarly successful results.

However, their approach encounters some problems when faced with multiple adjuncts of the type shown in (99), p.79. In attempting to translate Hans kocht oft gerne as Hans often likes cooking, their method translates it instead as Hans likes cooking often, which is wrong. Like the correspondence-based approach of Kaplan et al. (1989), often can only have wide scope over like if partial nodes can be rewritten to the two
terms kochen(n1) and SUBJ(n1, n2), which must be rewritten as the XCOMP of like, namely XCOMP(n4, n1), while ADJS(n1, n3), on the other hand, must be rewritten in such a way so as to attach to n4: ADJS(n4, n3).

As we saw previously with respect to (99), the same f-structure (109), p.80, represents both scoping possibilities, and as the approach of Dorna et al. is based on underspecified semantic representations, the underspecification is naturally preserved in the term language. The point is, however, that strings such as Hans kocht oft gerne are not ambiguous, so the effect that surface order has in restricting the meaning of such strings must be replicated so as not to hypothesize an ambiguity in transfer which is not there in the source, thereby leaving disambiguation to the generation component. They note that the semantics component reflects surface word-order in terms of its subordination constraints. For instance, the Hans often likes cooking reading is shown in (170):

$$l_\top : \begin{array}{c} x_2 : \text{Hans}(x_2) \end{array} \quad \ldots \quad l_\top : \begin{array}{c} x_2 : \text{Hans}(x_2) \\ \uparrow \end{array}$$

$$l_3 : \begin{array}{c} \text{oft}(l_3) \end{array} \quad \ldots \quad l_3 : \begin{array}{c} \text{often}(l_3) \\ \uparrow \end{array}$$

(170)

$$l_4 : \begin{array}{c} \text{gerne}(l_4) \end{array} \quad \ldots \quad l_4 : \begin{array}{c} \text{like}(x_2, l_4) \\ \uparrow \end{array}$$

$$l_1 : \begin{array}{c} \text{kochen}(x_2) \end{array} \quad \ldots \quad l_1 : \begin{array}{c} \text{cook}(x_2) \end{array}$$

Dorna et al. then observe that linear effects can be encoded in LFG using a set of f-precedence constraints, as in (171):

$$\begin{array}{c} \text{ADJS} \{ \begin{array}{c} \text{PRED } \alpha \end{array} l_2, \begin{array}{c} \text{PRED } \beta \end{array} l_3 \} \{ l_2 \prec_f l_3 \} \end{array}$$

(171)

Semantic subordination and f-precedence constraints are then linked to block the Hans kocht oft gerne ↔ Hans likes cooking often possibility with the result that headswitching and multiple adjunct interaction works properly.

Dorna et al. conclude by restating that the original f-structure to term rewriting described here suffers from the same problems of the correspondence-based approach of Kaplan et al. (1989), but these are overcome by doing transfer on UDRSSs. However, they note further that “semantics does not come for free, nor does it always blend as seamlessly with syntactic representation as one would hope for”. If semantics is not defined in the grammar, it will need to be encoded as a set of correspondences, depending on the system design.

### 3.3 Other Translation Problems

#### 3.3.1 Multiple Members of the Adjunct Set

Sadler et al. (1990) also show that the two approaches of Kaplan et al. (1989) cannot deal straightforwardly with translation examples where the head translates as head plus dependant, as in (172):
(172) a. ludothèque ←→ toy library
     b. vieux ludothèque ←→ old toy library

F-structures for the two NPs in (172b) are those in (173):

\[
\begin{align*}
(173) & \quad \left[ \begin{array}{c}
\text{PRED} \\
\text{ADJUNCT}
\end{array} \right. \\
& \quad \left[ \begin{array}{c}
\text{library}' \\
\{ [ \text{PRED} \quad \text{‘old’} ] \}
\end{array} \right] \\
& \quad \left[ \begin{array}{c}
\text{PRED} \\
\text{ADJUNCT}
\end{array} \right. \\
& \quad \left[ \begin{array}{c}
\text{ludothèque}' \\
\{ [ \text{PRED} \quad \text{‘vieux’} ] \}
\end{array} \right]
\end{align*}
\]

The first model of Kaplan et al. (op cit.) might attempt to translate (172a) using the regular \( \tau \)-equation for \textit{toy} in (174):

\[
(174) \quad \text{toy}:
\]

\( (\tau \uparrow \text{PRED FN}) = \text{miniature} \)

The head noun (175) would be translated using the ‘special’ \( \tau \) case for \textit{library} in (175):

\[
(175) \quad \text{library}:
\]

\[
\begin{align*}
\text{[PRED = toy]} \in \epsilon_c \uparrow \text{ADJ} \\
(\tau \uparrow \text{PRED FN}) = \text{ludothèque}
\end{align*}
\]

However, there are a number of problems with this approach. Firstly, adjuncts will be translated via \( \tau \)-equations annotated to the c-structure rule, as in (176):

\[
(176) \quad \text{NP} \rightarrow \text{AP}^* \quad \text{N}
\]

\[
\downarrow \in (\uparrow \text{ADJUNCT}) \\
\tau \downarrow \in (\tau \uparrow \text{ADJUNCT})
\]

The possibility of making such equations optional is not an avenue worth pursuing as it would lead to a general problem of non-translation of adjuncts. The alternative, as here, is to produce a target f-structure corresponding to \textit{ludothèque miniature} as the translation of \textit{toy library}. The problem is, therefore, one of attempting to specify the equations in such a way so that the adjunct remains ‘untranslated’ in these circumstances.

There is, however, a rather more serious problem with (174), namely that LFG specifically excludes reference to adjuncts in lexical entries:

“…since there is no notation for subsequently referring to particular members of that set (i.e. the set of adjuncts), there is no way that adjuncts can be restricted by lexical schemata associated with the predicate …Since reference to the adjunct via the ‘down arrow’ is not possible from other places in the string, our formal system makes adjuncts naturally context-free.”

(Kaplan & Bresnan, 1982:216)

Despite this, it might appear from Wedekind & Kaplan (1993) that indexing of members of the adjunct set \textit{is} nevertheless possible, so that we might be able to alter (173) into (177):

\[
99
\]
Now, together with lexical equations such as (175), we should be able to use restriction to stipulate the translation equation in (178):

\[(178) \quad \tau(\uparrow\text{ADJUNCT} \setminus \{b\}) = (\tau\uparrow\text{ADJUNCT})\]

That is, to translate the adjunct set, translate all of the adjuncts except that indexed \(b\). However, such indexation is not, in fact, possible. The indexing shown in Wedekind & Kaplan (1993) are just names attached to the structures to enable them to be referred to in the text (Kaplan, personal communication).

Another, more recently aired possibility which does enable access to the individual members of a set is an extension to functional uncertainty (Kaplan, personal communication). Here, the set-membership operator is allowed to be an element of an uncertainty path, so \((\uparrow\text{ADJUNCT} \in )\), for example, is a well-formed equation, just as \((\uparrow\text{COMP} \ast \text{SUBJ})\) is. In the latter case, the expression takes on values (non-deterministically) of any of the f-structure units that the path reaches, and the rest of the constraints then operate on that non-deterministic choice. The expression \((\uparrow\text{ADJUNCT} \in )\) denotes (non-deterministically) one particular element of the adjunct set. Therefore, if the set has two elements, there would be two disjunctive paths of computation, where the expression would denote the first of the elements on one path and the second on the other path.

Given this, for example (172a) one can envisage a solution with one correct and one wrong answer, as shown in (179):

\[(179) \quad \begin{align*}
&\text{a. toy library} \longrightarrow \text{ludothèque} \\
&\text{b. toy library} \longrightarrow ^*\text{ludothèque miniature}
\end{align*}\]

Worse still, there might be more than one wrong answer in the case of (172b), as shown in (180):

\[(180) \quad \begin{align*}
&\text{a. old toy library} \longrightarrow \text{vieux ludothèque} \\
&\text{b. old toy library} \longrightarrow ^*\text{vieux ludothèque miniature} \\
&\text{c. old toy library} \longrightarrow ^*\text{ludothèque} \\
&\text{d. toy library} \longrightarrow ^*\text{ludothèque miniature}
\end{align*}\]

Given this, it appears that the necessary context cannot be provided in the lexical entry for \textit{toy} for the translation of \textit{ludothèque} without violating LFG theory or without producing some wrong translations in conjunction with the correct one.

Sadler \textit{et al.} (1990) revise the notion and role of the transfer dictionary in LFG in order to capture such lexical dependencies without tuning f-structures. In the model of Kaplan \textit{et al.} (1989), translation equations come from either the source c-structure annotations or from a transfer lexicon. However, this lexicon is organized according to the source language grammar, i.e. it contains \(\tau\)-equations for the units of the source lexicon, so that these might be viewed as “just another projection” contained in the set of of monolingual
lexical equations (cf. (116), p.82, and resultant discussion). Sadler et al. (1990) consider this not to be right; there should instead be a proper transfer lexicon, organized around translationally relevant units, thereby viewing the source f-structure through the perspective of the target language, as in most rule-based MT systems.

The problem with cases containing adjuncts such as (172) is that we get multiple translations: for *toy library* both *ludothéque* and *ludothèque miniature* would be offered as translations, as shown in (179), and for *toy car* we would get both *voiture miniature* and *voiture*. Either the null translation or the regular translation of the adjunct need to be constrained where this is appropriate. We need to specify which adjunct serves as context in examples such as (173) and is to be ‘untranslated’, as in (181):

\[(181) \quad (\tau \uparrow ADJUNCT \ PRED \ toy) = \text{NIL} \]

Alternatively, this may be achieved via (182):

\[(182) \quad \tau(\uparrow ADJUNCT \ PRED \ toy \ PRED \ FN) = \text{NIL} \]

Either of these is inconsistent with the regular translation of *toy* (as *miniature*). However, Sadler et al. (op cit.) note that this can be simulated by structuring the transfer lexicon further to allow conjoined entries, such as (183):

\[(183) \quad \left[ \begin{array}{c} \text{library: } (\tau \uparrow \text{PRED FN}) = \text{ludothèque} \\ \land \text{toy: } (\tau \uparrow) = \text{NIL} \end{array} \right] \]

### 3.3.2 Anaphoric Dependencies

The correspondence-based approach to LFG-MT (Kaplan et al., 1989) fails to treat anaphoric dependencies in translation correctly. Sadler & Arnold (1992) show one example where the approach of Kaplan et al. works correctly, namely (184):

\[(184) \quad \text{The man who I saw} \rightarrow \text{L’homme que j’ai vu} \]

With English as the source language, this requires the rules in (185):

\[(185) \quad \begin{align*}
\text{NP} & \rightarrow \text{NP} \quad \quad S' \\
(\uparrow \text{RELMOD}) & = \downarrow \\
\tau(\uparrow \text{RELMOD}) & = (\tau \uparrow \text{RELMOD}) \\
\tau(\downarrow \text{RELTOPIC}) & = (\tau \downarrow \text{RELTOPIC}) \\
S' & \rightarrow \text{XP} \quad \text{S} \\
(\uparrow \text{RELTOPIC}) & = \downarrow \\
(\uparrow \{XCOMP,COMP\} * \text{GF}) & = \downarrow 
\end{align*} \]

In addition, we need the lexical entry for *see* in (127), p.86, as well as the entry for *who* in (186):
(186) \( \text{who}: N \)
PREDF= 'who'
HUMAN+=
\((\tau \mapsto \text{PREDF FN}) = \text{`QUE'}\)

‘QUE’ subsumes all its possible variants. The f-structures corresponding to the NPs in (184) are those in (187):

\[
\begin{align*}
\text{(187)} & \\
\text{RELMOD} & : e_1 & \\
\text{RELTOPIC} & e_2 & \text{PREDF} \quad \text{'who'} \]
\end{align*}
\]

In this simple example, the desired re-entrance falls out quite straightforwardly given the rules in (185) and the lexical entries in (127) and (186), as the relevant constraints in (188) show:

\[
\begin{align*}
\text{(188)} & \\
(f_1, \text{RELTOPIC}) &= \tau(e_1, \text{RELTOPIC}) = \tau(e_2) \\
(f_1, \text{OBJ}) &= \tau(e_1, \text{OBJ}) = \tau(e_2)
\end{align*}
\]

However, Sadler & Arnold state that “this approach is only capable of producing intuitively correct structures in cases where the conditions on unbounded dependencies are parallel in the source and target languages”, as here. Kaplan et al. (1989) give an example using unbounded dependencies, namely (189):

\[
\begin{align*}
\text{(189)} & \\
\text{The letter which I have answered} & \rightarrow \text{La lettre à laquelle j’ai répondu}
\end{align*}
\]

Sadler & Arnold show that rather than producing the correct translation in (189), what is actually produced given the rules and lexical entries in Kaplan et al. (1989) is the f-structure in (190):

\[
\begin{align*}
\text{(190)} & \\
\text{RELMOD} & : f_1 & \\
\text{RELTOPIC} & f_2 & \text{PREDF} \quad \text{`QUE'} \\
\text{SUBJ} & f_3 & \text{PREDF} \quad \text{`je'} \\
\text{OBL}_{go} & f_2 & \text{PREDF} \quad \text{`à'}
\end{align*}
\]

This leads to the derivation of the ungrammatical (191), as prepositions cannot be stranded in French:

\[
\begin{align*}
\text{(191)} & \\
\text{*La lettre laquelle j’ai répondu à} & \rightarrow \text{The letter which I have responded to}
\end{align*}
\]

The problem here is that the relativized positions in the two languages are not identical. What happens in practice is that the source grammar dictates what will be re-entrant in the target structure.

Sadler & Arnold suggest a more flexible method of handling such variation across languages using underspecified constraints. The problem with (191) is that separate \( \tau \)-equations are given for both the source RELTOPIC and ‘within-clause’ function paths involved in the re-entrance, but both equations have the same value. Thus a target re-entrance is produced automatically whose value is the translation of the source

102
object that both equations point at. In (191) this results in an incorrect target f-structure. What Sadler &
Arnold propose to do is to restrict the τ-correspondence to just one of the paths contributing to the source
re-entrance, which allows a constraint-based solution to examples such as (189).
Sadler & Arnold point out that Kaplan et al. (1989) choose exactly this strategy to solve differences in
control, as in likely ↔ probable (cf. (81), p.73), where no τ-correspondence is given for the SUBJ of likely,
but a τ-equation is given for the re-entrant SUBJ in the complement clause. Therefore, on the French side
the grammar introduces an expletive SUBJ for probable and no re-entrance is postulated at all, as required.
Nevertheless, Kaplan et al. do not carry this technique over to the treatment of unbounded dependencies.
Sadler & Arnold state that if no τ-equations are placed on rules containing RELTOPICs, and a τ-correspondence
is stated only over the path terminating in the within-clause function, then the target f-structure in (192) is
produced:

\[
(192) \quad \text{RELMOD} \quad f_1 \quad \begin{array}{c}
\text{PRED} \\
\text{SUBJ} \\
\text{OBL}_{go}
\end{array} \begin{bmatrix}
\text{répondre}((\uparrow \text{SUBJ})(\uparrow \text{OBL}_{go}))' \\
\text{je}' \\
\text{à'} \\
\text{QUE}'
\end{bmatrix}
\]

(192) is the same as (190) except for the RELTOPIC function. The (ungrammatical) string that (192)
corresponds to is j’ai répondu à laquelle. What needs to be done to produce an f-structure corresponding
to the French string in (189) is to ensure that an appropriate value for the RELTOPIC function is given.
Sadler & Arnold suggest that there are three sources of potentially valuable information available:

1. There is the solution to the functional uncertainty equation associated with the XP node in (185),
   which in the case of (189) is (↑ RELTOPIC) = (↑ OBJ). Quite straightforwardly, a τ-equation such
   as (193) can be produced:

\[
(193) \quad (\tau \uparrow \text{RELTOPIC}) = \tau(\uparrow \text{OBJ})
\]

Let us assume the lexical entry for answer in (194):

\[
(194) \quad \text{answer, V: }
\begin{array}{c}
\uparrow \text{PRED} = \text{answer}((\uparrow \text{SUBJ})(\uparrow \text{OBJ}))' \\
\tau \uparrow \text{PRED FN} = \text{répondre}((\uparrow \text{SUBJ})(\uparrow \text{OBL}_{go}))' \\
\tau(\uparrow \text{SUBJ}) = (\tau \uparrow \text{SUBJ}) \\
\tau(\uparrow \text{OBJ}) = (\tau \uparrow \text{OBL}_{go} \text{ OBJ})
\end{array}
\]

If (194) is applied to (185), (195) is derived:

\[
(195) \quad (\tau \uparrow \text{RELTOPIC}) = \tau(\uparrow \text{OBL}_{go} \text{ OBJ})
\]

This can be achieved by adding (τ ↑ RELTOPIC) = τ(↑ α) for every solution α of the uncertainty on
the right-hand side of the functional uncertainty equation in (185). (195) cannot be added to the set
of other τ-equations as it would re-establish the unwanted re-entrancy in (190) that gives rise to (191).
2. The target grammar will ensure that if RELTOPIC is present, then some path within the RELTOPIC attribute contains WH=+, to prevent wh-fronting of an XP that does not contain a WH-phrase. Sadler & Arnold suggest that this equation would be as in (196):

\[(196) \quad (\uparrow RELTOPIC\{OBJ,POSS}\ast WH) =_c +\]

3. The target grammar contains an equation such as (197):

\[(197) \quad (\uparrow RELTOPIC) = (\uparrow \{COMP,XCOMP\} \ast \{SUBJ,OBJ,\text{OBL}^{th}\})\]

(197) is required in order to establish a relation between the RELTOPIC and some within-clause function. Restricting identity between the RELTOPIC and the smaller set of GFs in (197) prevents the preposition stranding in (191).

(195) stipulates that there must be a RELTOPIC attribute on the target side. Naturally, if the equation derived from the source is consistent with the constraints in the target grammar, then it is chosen. If not, Sadler & Arnold state that the ‘closest possible solution’ is to be found by comparing the constraint with the functional uncertainty equations for the target language. For relative clauses, there are two such equations: (196) requires that RELTOPIC contain a WH=+ path, and (197) expresses the re-entrance between the RELTOPIC and some within-clause function. In the case of (185), the solution is (198):

\[(198) \quad (\uparrow \text{OBL}_{go}) = (\uparrow RELTOPIC) \quad (\uparrow RELTOPIC\ OBJ\ WH) = +\]

This pair of equations enables the correct f-structure in (199) to be built:

\[(199) \quad \text{RELMOD} \begin{bmatrix} f_1 \text{ RELTOPIC } f_2 \text{ PRED ‘à’ OBJ } \text{ PRED ‘QUE’ WH } + f_3 \text{ PRED ‘répondre((\uparrow SUBJ)(\uparrow \text{OBL}_{go}))’ SUBJ } f_3 \text{ OBJ } f_2[] \text{ OBL}_{go} \end{bmatrix}\]

Sadler & Arnold state that this sort of solution represents a genuine advantage of constraint-based approaches over rule-based methods. Structural approaches require a check that all source items have been translated. In this case, this would mean writing a rule which translates the value of the RELTOPIC function as ‘nil’, in effect deleting it on the target side. Furthermore, the transfer rules would output a structure like man [ [] I have seen who], and rules would be required to link who with the RELTOPIC position ([]). Such rules will not necessarily exist, particularly if they are associated with c-structure rules, as there are no empty categories in LFG c-structures on which the relevant equations could be placed. The constraint-based approach, on the other hand, enables various parts of the translation relation to be underspecified, allowing monolingual and bilingual constraints to interact in a flexible manner to produce the correct results.
3.4 Generation in LFG

Approaches to LFG-MT such as that of Kaplan et al. (1989) provide $\tau$-equations to derive target f-structures from input source language f-structures (or s-structures). In order to generate target strings, we require a process which produces a target c-structure from a target f-structure (or s-structure), from which the string can be trivially read off. A number of approaches to LFG generation exist, and we shall describe each of these in the following sections.

3.4.1 Structure-Driven Generation

Wedekind (1988) provides separate algorithms which enable generation from f-structures and s-structures. Both are based on Wedekind’s (1985) proposal which converts LFG into a monostatical theory of language, based on the observation that c-structure provides little else than constraints on surface word-order. Despite this, Wedekind notes that both algorithms are sufficiently general to be applicable to other unification-based formalisms.

Generation from F-structures

Wedekind (1988) provides an algorithm which generates terminal strings from f-structures. This algorithm must for every LFG define a relation $\Gamma_{\phi}(\Phi, s)$ ($s$ is *generable* from $\Phi$) between DAGs and terminal strings, as in (200):

$$(200) \quad \forall \Phi \in \text{DAG} \ \forall s \in V^{T}(\Gamma_{\phi}(\Phi, s) \iff C(\ldots \Phi, s \ldots))$$

$C(\ldots \Phi, s \ldots)$ defines the ‘adequacy condition’ for LFG-generation. This may ideally state that “$s$ is derivable from the start symbol $S$ and $s$ has f-structure $\Phi$” (*op cit*, p.732), meaning that a generator for LFG $G$ must accept an input structure $\Phi$ by building a string $s$, if $s$ is derivable with f-structure $\Phi$ in $G$. In turn, this implies that the generator will produce no output if the input f-structure is not well-formed.

The CFG rules produce a c-structure for a string, the nodes of which are then mapped via $\phi$ to the nodes in $\Phi$, the unique minimal f-structure which satisfies the c-structure annotations. Finally, all constraints in the f-description are checked and $\Phi$ is checked for completeness and coherence. Wedekind observes that if the f-structure is built after the c-structure, any functional information contained in an input structure cannot be used to control the derivation of the c-structure. “A decidable generation procedure presupposes the possibility of comparing the input structure with the partial f-structure of a derived c-structure. Thus, in order to drive the c-structure derivation by a given input structure it is necessary to derive the partial f-structure in parallel to a partial c-structure” (*ibid*). Assuming Wedekind’s (1986) monostatical version of LFG, a derivation is a sequence of quadruples $<c, \Phi, \phi, C^\phi>$ where $c$ is a partial c-structure, $\Phi$ is a partial f-structure, $\phi$ is a mapping from c-structure nodes into the set of nodes of $\Phi$, and $C^\phi$ is a set of constraints. We slightly adapt a simple example which Wedekind gives to show how this algorithm might work. Consider the c-structure rules in (201):
Figure 3.1: Interim Derived Quadruple using F-structure Driven Generation

\[
S \rightarrow \text{NP} \quad \text{VP} \\
\quad (\uparrow \text{SUBJ}) = \downarrow \quad \uparrow = \downarrow
\]

(201)

\[
\text{VP} \rightarrow \quad \text{V} \quad \text{VP}' \\
\quad \uparrow = \downarrow \quad (\uparrow \text{XCOMP}) = \downarrow
\]

These rules represent the relation in (202):

Let us assume the rule for the verb tries in (203):

\[
\text{V} \rightarrow \text{tries} \\
\quad (\uparrow \text{PRED}) = \text{"tries(\uparrow \text{SUBJ})(\uparrow \text{XCOMP})"} \\
\quad (\uparrow \text{SUBJ}) = (\uparrow \text{XCOMP SUBJ}) \\
\quad (\uparrow \text{NUM}) =_c (\uparrow \text{SUBJ NUM})
\]

(203) is represented by the quadruple in (204):

Given leaf V_2.1 in (202), the representation of V-rule (203) in (204) can now be applied to (201) to form the new object in Figure 3.1. The derived quadruple \(\langle \epsilon', \phi', \Psi', C^{\phi'} \rangle\) consists of a c-structure which is the result of expanding leaf 2.1 by \(c_r\), and a partial f-structure \(\Phi'\), which is the minimal extension of \(\Phi\) by unifying the DAG \(\Phi\) with the DAG \(\Phi_r\) introduced by rule (203). Note also that the constraint set has been updated by attaching 2.1 as a prefix to each node index within the constraint \{\(\emptyset \text{NUM} =_c (\emptyset \text{SUBJ NUM})\). Continuing in this fashion will result in a complete f-description for the f-structure of the sentence.

106
Given such a process, Wedekind notes that “we could use this derivation concept for generation ... However, such a concept of generation would not satisfy the adequacy condition” (Wedekind 1988:733). The problem is that while we can check for completeness and coherence, additional adjuncts could be introduced, or some of the original adjuncts might not be derived. What needs to be checked is that all paths and re-entrancies are derived, and that the derived structure is subsumed by the input structure ($\Phi_i \subseteq \Phi_n$). In this case it can be guaranteed that the f-structure of the generated string is the unique minimal structure that satisfies the constraints expressed in the input structure.

Wedekind checks that all paths are derived by distinguishing generator rules with grammar rules by prefacing all nodes in the DAG introduced by a rule (except for the root) with a ‘+’ marker. He observes that “if the root of the unordered tree $\Phi_i$ is also +-labelled and all nodes of the structure that is derived from $\Phi_i$ are +-labelled, then all paths of the input structure are derived” (ibid). Checking for reentrancies is controllable if the input structure $\Phi_i$ is unfolded: if the derived structure and the input structure are isomorphic, then all re-entrancies are derived. Checking the subsumption relation is enforced at each step.

**Generation from S-structures**

Wedekind (op cit.) also provides an algorithm which enables strings to be derived from semantic structures. He notes that ‘a semantic structure alone, however, is usually regarded as too poor with respect to the syntactic information relevant for “adequate generation results”’ (op, cit., p.734). Hence Wedekind uses syntactic f-structure information to drive the derivation via semantic structures, which is possible if one assumes that f-structures and s-structures are built up in parallel (cf. (208) below).4

To give an example of what ‘f-structure driven semantic generation’ means, let us assume the s-structure in (205):

\[
\begin{bmatrix}
\text{REL} & \text{kick} \\
\text{ARG1} & \begin{bmatrix} \text{REL} & \text{john} \end{bmatrix} \\
\text{ARG2} & \begin{bmatrix} \text{REL} & \text{peter} \end{bmatrix}
\end{bmatrix}
\]

Many different strings could in principle be derived from such a structure, depending on what syntactic information is available. For example, two sets of constraints might be those in (206):

\[
C_1^\phi \oplus \sigma = \left\{ \begin{array}{l}
(\emptyset \text{ SUBJ } \sigma) =_c (\emptyset \sigma \text{ ARG2}) \\
(\emptyset \text{ BYOBJ } \sigma) =_c (\emptyset \sigma \text{ ARG1})
\end{array} \right\}
\]

\[
C_2^\phi \oplus \sigma = \left\{ \begin{array}{l}
(\emptyset \text{ SUBJ } \sigma) =_c (\emptyset \sigma \text{ ARG2}) \\
(\emptyset \text{ BYOBJ } \sigma) =_c (\emptyset \sigma \text{ ARG1}) \\
(\emptyset \text{ BYOBJ } \sigma) =_c (\emptyset \text{ TOPIC})
\end{array} \right\}
\]

Note here that some constraints are purely syntactic, while others are what Wedekind terms ‘inter-modular’. The two strings which would be derived from these constraints are those in (207):

---

4cf. chapter 6, note 9, for discussion on the possible need for f-structure constraints to be dropped in order to derive a target string.
(207) a. \( C_1^\phi \oplus \sigma \Rightarrow \text{Peter is kicked by John.} \)
    
b. \( C_2^\phi \oplus \sigma \Rightarrow \text{By John, Peter is kicked.} \)

In later work, Wedekind (1999) determines that semantic-driven generation with unification grammars such as LFG or PATR is undecidable. However, Wedekind (1995) showed that the generation problem \( \Delta_G \) of a word string \( w \) from an f-structure \( \Phi \) is decidable \textit{iff} unification grammar \( G \) assigns \( \Phi \) to \( w \). Given structures such as (208), it is clear that semantic structures SEM subsume the f-structure \( \Phi \):

\[
(208) \begin{align*}
\text{PRED} & \quad \text{`swim(\langle \uparrow \text{SUBJ} \rangle)'} \\
\text{TENSE} & \quad \text{PRES} \\
\text{SUBJ} & \quad \begin{bmatrix} 
\text{PRED} & \text{`John'} \\
\text{REL} & \text{swim} \\
\text{ARG1} & \text{john} 
\end{bmatrix} \\
\text{SEM} & \quad \end{align*}
\]

Generation from a semantic structure \( \Phi' \) is decidable \textit{iff} \( \Phi' \) subsumes \( \Phi \). Wedekind (1999:277) describes this formally as (209):

\[
(209) \quad \exists w \exists \Phi (\Phi' \sqsubseteq \Phi \land \Delta_G (w, \Phi))
\]

Wedekind observes that this \textit{must} be decidable given that only a finite number of structurally related f-structures \( \Phi \) have to be tested for \( \{ w \mid \Delta_G (w, \Phi) \} \), but admits that “it is far from evident yet, how this structural relation is realized in natural language grammars” (\textit{op cit.}, p.281). Nevertheless, he notes that the solution may be found by discovering “a proportion \( k \) which bounds the size of an f-structure assigned to a string by the size of its subsuming semantic representation ... This would force the f-structures of the surface realizations of a semantic representation ... to be included in a finite and computable set of structurally related f-structures” (\textit{ibid}.

3.4.2 Generation from Under- and Overspecified Structures

Kohl (1992) extends Wedekind’s (1988, 1990) algorithm in two ways. His algorithm does not require that substructures which cannot be altered during the generation process be marked as such, and furthermore he considers that the requirement that partial feature structures be complete and coherent may be too restrictive.

The projects in which Kohl’s algorithm is used both involve feature structures which are underspecified with respect to the target grammar, meaning that the generator needs on some occasions to add information in order to derive structures which are valid with respect to the target grammar. Furthermore, input structures can be overspecified and/or ill-formed with respect to the target grammar. In the first instance, certain attribute-value pairs will need to be ignored, and in the latter some error-handling processes are required.

In terms of MT, we can expect mismatches between source and target structures fairly regularly: for example, translating from English into German, we might expect the source f-structure for the sentence \textit{John is
to contain an attribute ASPECT with value progressive. The target grammar would ignore this information as it is irrelevant for German: all we require is the translation Johannes schwimmt. If the source and target languages were reversed, however, we can suppose that unless the English generation component can add information to the input structure, then only the string John swims will be produced.

Kohl’s algorithm is head-driven, in that the semantic head is selected automatically, even if it is embedded in some substructure. This, he notes, “allows to generate (sic) in cases of head-switching” (1992:691). He contrasts his approach with that of Shieber et al. (1990) which “seems to be a bit problematic for grammars which describe head-switching phenomenons (sic)” (ibid), given that their approach requires precomputation of nodes which contain the semantic head.

### 3.4.3 Ambiguity-preserving Generation

Wedekind & Kaplan (1996:556) observe that natural language ambiguity ‘presents a particular challenge for machine translation’. As an alternative strategy, they wonder whether ambiguity-preserving generation using unification grammars is a decidable problem. For instance, rather than having to disambiguate I saw the man with the telescope, translations may be achievable in certain languages (French and German, say) which preserve the same ambiguities. However, their results are negative on the whole: “the problem of ambiguity-preserving generation (and thus ambiguity-preserving translation) is unsolvable even if the languages are described by unification grammars for which the parsing and generation problems separately are computable” (op. cit., p.557). Nevertheless, they conclude that empirical studies may show that ambiguity-preserving translation is enabled for specific constructions in certain language pairs.

### 3.4.4 Discussion

Kohl (1992:686) notes that in MT, “the grammars used for parsing and for generation are basically specific for the two languages one wants to translate between. It is usually desirable to specify [the target feature structure] only in as rudimentary and as general manner (sic) as possible. This means the details of how to generate a valid surface string of the target language are only known in the target grammar”. This statement applies in general specifically to transfer-based systems only: the Rosetta (Rosetta, 1994) system takes a different approach entirely. Perhaps the most important aspect of Rosetta is based on the observation that “(in) interlingual approaches ... the structures that are yielded by the analysis module must be accepted by the generation module”. This is a problem for all systems, except that in transfer-based systems it is the output of transfer, rather than analysis, which is input into generation. We discussed some of these issues with respect to the subset problem and tuning in section 1.3.2.

### 3.5 Summary

In sum, the firm mathematical foundations of LFG provide the linguist with a powerful yet intuitive set of tools which facilitate the description of natural languages both monolingually and multilingually. The correspondence-based approach to LFG-MT of (Kaplan et al., 1989) is able to explicitly relate exactly those
parts which play the decisive role in translation via its \( \tau \)-equations.

Nevertheless, we have described certain cases of headswitching, as well as cases with multiple adjuncts, that LFG-MT still appears not to be able to handle. Despite attempts at solving these problematic constructions using approaches based on linear logic and restriction, further problems are introduced which require a solution.

We shall show in chapter 6 that as well as being able to handle ill-formed and unknown strings, LFG-DOT, a model of translation based on LFG-DOP, is able to solve some of these problematic cases, as well as preferring the specific translation over the compositional, default alternative. The same may be seen in chapter 5 with respect to the DOT2 model of translation. Our LFG-DOT models propose alternatives to the techniques described in section 3.4 for generating the target string, by incorporating a target language LFG-DOP model. The following chapter describes two LFG-DOP models on which LFG-DOT is based.
Chapter 4

LFG-DOP: A New, Hybrid Architecture for NLP

Bod & Kaplan (1998, 1999) have recently augmented DOP with the syntactic representations of Lexical Functional Grammar to create a new, more powerful hybrid model of language processing, LFG-DOP.

4.1 Opportunities for Hybridity: LFG-DOP

Tree-DOP has been used to perform experiments on the Penn Treebank\(^1\) and the OVIS\(^2\) (Dutch Public Transport Information System) corpus (Bod 1993, 1995; Sima’an 1995, 1996, 1999; Bonnema et al., 1997) which show an increase in parse accuracy when larger, more complex tree fragments are considered. Such approaches are necessarily limited to those contextual dependencies actually occurring in the corpus, which is a reflection of surface phenomena only. It has been known for some time that purely context-free models are insufficiently powerful to deal with all aspects of human language. In this regard, DOP models have been augmented (van den Berg et al., 1994; Tugwell 1995) to deal with richer representations, but such models have remained context-free. LFG, however, is known to be beyond context-free. It can capture and provide representations of linguistic phenomena other than those occurring at surface structure. With this facility in mind, the functional structures of LFG have been allied to the techniques of DOP to create a new model, LFG-DOP (Bod & Kaplan, 1998, 1999), which adds a measure of robustness (both with respect to unseen as well as ill-formed input) not available to models based solely on LFG.

4.1.1 The Original LFG-DOP Model

As with Tree-DOP, LFG-DOP needs to be defined using the same four parameters outlined in chapter 2. Its representations are simply lifted \textit{en bloc} from LFG theory, so that each string is annotated with a

\(^1\)http://linc.cis.upenn.edu/~treebank/home.html
\(^2\)http://grid.let.rug.nl:4321/
c-structure, an f-structure, and a mapping \( \phi \) between them. Well-formedness conditions operate solely on f-structure, as usual.\(^3\)

Since we are now dealing with \((c,f)\) pairs of structure, the Root and Frontier decomposition operations of DOP need to be adapted to stipulate exactly which c-structure nodes are linked to which f-structure fragments, thereby maintaining the fundamentals of c- and f-structure correspondence. Given that LFG c-structures are little more than annotated PS-trees allows us to proceed very much on the same lines as in Tree-DOP. Root erases all nodes outside of the selected node, and in addition deletes all \( \phi \)-links (informally, parts of the f-structure linked to a c-structure node) leaving the erased nodes, as well as all f-structure units that are not \( \phi \)-accessible from the remaining nodes. Bod & Kaplan (op cit.) define \( \phi \)-accessibility thus:

\[
\text{"An f-structure unit } f \text{ is } \phi \text{-accessible from a node } n \text{ iff either } n \text{ is } \phi \text{-linked to } f \text{ (that is, } f = \phi(n)\text{) or } f \text{ is contained within } \phi(n) \text{ (that is, there is a chain of attributes that leads from } \phi(n) \text{ to } f\)."
\]

As an example, consider (210):

\[
\begin{align*}
\text{NP}_n & \quad \text{VP}_m \\
\text{John} & \quad \text{likes} & \text{Mary} \\
\phi & \quad f_1 & \quad f_2 & \quad f_3 & \quad f_4 & \quad f_5 \\
\text{PRED} & \quad \text{LIKE} & \quad \text{OBJ} & \quad \text{NUM} & \quad \text{SG} & \quad \text{PERS} & \quad \text{3} \\
\text{TENSE} & \quad \text{PRESENT} & \quad \text{PASSIVE} & \quad -
\end{align*}
\]

The \( \phi \)-links are shown in (211):

\[
\begin{align*}
\phi(n_1) &= f_1 \\
\phi(n_2) &= f_2 \\
\phi(n_3) &= f_3 \\
\phi(n_5) &= f_5 \\
\phi(n_4) &= f_4 \\
\phi(n_1) &= \phi(n_3) = \phi(n_4)
\end{align*}
\]

\( \phi \)-accessibility reflects the intuitive notion that nodes in a tree carry information only about the f-structure elements to which the root node of the tree permits access,\(^4\) as in (212):

\[\text{...}\]

---

\(^3\)Except where we use competition sets 0 and 1 (described below) of Bod & Kaplan (1998) as input into our probability models.

\(^4\)Bod and Kaplan (1999:12) state that they “follow the more recent proposal of Zaenen & Kaplan (1995) in which lexical heads are directly linked to semantic forms and not to their enclosing f-structures, while other primitive feature values remain unlinked”. In the original exposition of LFG (Kaplan & Bresnan, 1982), the lexical heads would also be \( \phi \)-linked to the f-structures whose semantic-form PRED values they provide. Thus it seems clear that adopting the approach of Zaenen & Kaplan (1995) cuts down on the possible number of LFG-DOP fragments.
In (212), all the f-structure units are \( \phi \)-accessible from the S and VP nodes, but TENSE and the top-level PRED (the main verb *swim*, here) cannot be accessed via \( \phi \) from the subject NP node.

*Frontier* operates as in Tree-DOP, deleting all subtrees of the selected frontier nodes. Furthermore, it deletes all \( \phi \)-links of these deleted nodes together with any semantic form corresponding to the same nodes, as in (213):

This illustrates the ability of *Root* nodes to access certain features (TENSE, here) even after subnodes have been deleted. While this may look odd to speakers of English, Bod and Kaplan (1998) note that subject-tense agreement *is* seen in some languages (such as Hindi). Consequently, there is no universal principle which rules out fragments such as (213).

It is, however, possible to prune (213) still further, as (214) illustrates:

This is achieved by applying a third, and new operation, *Discard*, to the TENSE feature in (213). This represents directly the probability that there is no subject-tense dependency in English. Consequently we would expect to see such fragments more frequently in our treebanks. It is this *Discard* operation that adds considerably to LFG’s robustness (cf. section 4.2 below). *Discard* provides generalized fragments from those derived via *Root* and *Frontier* by freely deleting any combination of attribute-value pairs from an f-structure except those that are \( \phi \)-linked to some remaining c-structure node, or that are governed by the local predicate. Its introduction also necessitates a new definition of the grammaticality of a sentence with respect to a corpus, namely any sentence having at least one derivation whose fragments are produced only by *Root* and *Frontier* and not by *Discard*. Note that unless non-*Discard* fragments are kept apart from *Discard* fragments (a solution we advocate in section 4.5), these latter fragments will need to be marked in some way so as to facilitate the consideration of grammaticality.
Figure 4.1: The complete LFG-DOP treebank (no Discard) for the sentence John swims
The complete LFG-DOP treebank (ignoring for the time being the effects of the Discard operator) for the sentence John swims is given in Figure 4.1. Note that it is not the case that each c-structure fragment in an LFG-DOP corpus is linked to a unique f-structure fragment. As an example, we show in (215) three fragments from Figure 4.1:

\[
S 
\begin{array}{c}
NP \\
\bigg) \\
V \\
\end{array} 
\begin{array}{c}
VP \\
\bigg) \\
V \\
\end{array} 
\begin{array}{c}
V \\
\bigg) \\
swims \\
\end{array} 
\begin{array}{c}
V \\
\bigg) \\
swims \\
\end{array} 
\]

(215)

The c-structure fragments in (215) all map to the same f-structure fragment, namely (216), because of equations such as $\phi(n_1) = \phi(n_3) = \phi(n_4)$ in (210):

\[
\begin{array}{c}
\text{SUBJ} \\
\text{NUM} \\
\text{SG} \\
\end{array} 
\begin{array}{c}
\text{PRED} \\
\text{swim}(\langle \uparrow\text{SUBJ} \rangle) \end{array} 
\begin{array}{c}
\text{TENSE} \\
\text{PRES} \\
\end{array} 
\]

(216)

Note also that two of the fragment pairs produced by Discard in Figure 4.2, shown here in (217), illustrate the relaxation of the SUBJ=SG constraint with swims:

\[
\begin{array}{c}
\text{VP} \\
V \\
\end{array} 
\begin{array}{c}
\text{SUBJ} \\
\text{[]} \\
\end{array} 
\begin{array}{c}
\text{PRED} \\
\text{swim}(\langle \uparrow\text{SUBJ} \rangle) \end{array} 
\begin{array}{c}
\text{TENSE} \\
\text{PRES} \\
\end{array} 
\]

(217)

\[
\begin{array}{c}
\text{VP} \\
V \\
\end{array} 
\begin{array}{c}
\text{SUBJ} \\
\text{[]} \\
\end{array} 
\begin{array}{c}
\text{PRED} \\
\text{swim}(\langle \uparrow\text{SUBJ} \rangle) \end{array} 
\begin{array}{c}
\text{TENSE} \\
\text{PRES} \\
\end{array} 
\]

Whilst this may be counter-intuitive for this example, it would be rather more normal to omit the value of the SUBJ’s NUM for past tense verbs or modals (for English). Notwithstanding the presence of fragments such as (217), we report in section 4.2 that LFG-DOP nevertheless shows a preference for SUBJ=SG over SUBJ=PL or undefined NUM for such sentences. Regarding (217), Bod & Kaplan (1999:37) provide the interesting observation that the specification of Discard “reflects the fact that LFG representations, unlike LFG grammars, do not indicate unambiguously the c-structure source (or sources) of their f-structure feature values”.

Composition is also a two-step operation. C-structures are combined by leftmost substitution, as in Tree-DOP, subject to the matching of their nodes. F-structures corresponding to these nodes are then recursively unified, and the resulting f-structures are subjected to the grammaticality checks of LFG.
Finally, $P(f \mid CS)$ denotes the probability of choosing a fragment $f$ from a competition set $CS$ of competing fragments. The probability of an LFG-DOP derivation is the same as in Tree-DOP (cf. (27), p.36): it is just the derivation itself which changes. In (28)-(33) for Tree-DOP, pp.38-38, we were interested in choosing a tree $t$ from a treebank: now we are interested in selecting a $\langle c, f \rangle$ pair from a corpus. Similarly then, an LFG-DOP derivation is also produced by a stochastic branching process which at each stage in the process randomly samples from a competition set $CS$ of competing samples. Here $CP(f \mid CS)$ denotes the probability of selecting a fragment pair $f$ from a competition set $CS$ containing $f$. The probability of an LFG-DOP derivation $D = \langle f_1, f_2, \ldots f_n \rangle$ can be expressed as (218):

$$P(\langle f_1, f_2, \ldots f_n \rangle) = \prod_{i=1}^{n} CP(f_i \mid CS_i)$$

This competition probability $CP(f \mid CS)$ is expressed in terms of fragment probabilities $P(f)$ in (219):

$$CP(f \mid CS) = \frac{P(f)}{\sum_{f' \in CS} P(f')}$$

In Tree-DOP, apart from the $Root$ and $Frontier$ operations, there are no other well-formedness checks. LFG, however, has a number of grammaticality conditions, some of which—the Completeness check at least—cannot be evaluated during the stochastic process. Given this, probabilities for valid representations can be defined by sampling post hoc only from such representations as are output from the stochastic process. The probability of sampling a valid representation is (220):

$$P(R \mid R \text{ is valid}) = \frac{P(R)}{\sum_{R' \text{ is valid}} P(R')}$$

Bod & Kaplan (op cit.) note that (220) assigns probabilities to valid representations whether or not the stochastic process guarantees validity. The valid representations for a particular utterance $u$ are obtained by a further sampling step, with their probabilities given by (221):

$$P(R \mid R \text{ is valid and yields } u) = \frac{P(R)}{\sum_{R' \text{ is valid and yields } u} P(R')}$$

Comparing (220)-(221) with the equivalent formula for calculating the probability of a particular analysis for a Tree-DOP representation (36), p.41, we note that the LFG-DOP formulae contain references to valid structures. In Tree-DOP, apart from the root-matching criterion, there are no other validity conditions in the stochastic process. In LFG-DOP, depending on the competition set chosen, there may be several.

If we choose to enforce the LFG grammaticality checks at various stages in the process, a number of different competition sets are produced. Bod & Kaplan (1998) describe three such sets linked to the probability models for LFG-DOP:
1. This is a straightforward extension of the Tree-DOP probability model, M1, where the choice of a fragment depends only on its Root node and not on the Uniqueness, Completeness or Coherence conditions of LFG, which are enforced off-line. In a similar way to Tree-DOP (31), the competition set for the $n^{th}$ step using model M1 is (222):

$$(222) \quad CS_n = \{ f : \text{root}(f) = LNC(F_{n-1}) \}$$

This assumes that $F_{n-1} = f_1 \circ f_2 \circ \ldots \circ f_{n-1}$ is the sub-analysis immediately prior to the $n^{th}$ step in the whole process, and $LNC(F_{n-1})$ denotes the leftmost non-terminal category on the frontier of $F_{n-1}$, and root($f$) denotes the root category of the c-structure fragment of $f$.

As with Tree-DOP (32), p.38, this competition set depends only on the leftmost non-terminal category of the c-structure tree of the current fragment, giving the competition probability for an LFG-DOP fragment as (223):

$$\quad \quad \quad \quad \quad \quad (223) \quad CP(f \mid CS) = CP(f) = \frac{P(f)}{\sum_{f' : \text{root}(f') = \text{root}(f)} P(f')}$$

Consequently, unless a large number of valid representations are sampled with high conditional probabilities, this model can be expected to produce many invalid representations. As shown in (220)-(221), LFG-DOP relies on sampling valid structures, which will entail a fair amount of processing to extract these from the set of all representations produced.

2. C-structure nodes must match, and f-structures must be unifiable if two LFG fragments are to be combined. This model M2 takes the LFG Uniqueness condition (namely that each attribute has only one value) as well as the Root category into account. As the resultant fragments produced vary depending on the derivation followed, unifiability must be determined at each step in the process. For M2, the competition set for the $n^{th}$ step is (224):

$$(224) \quad CS_n = \{ f : \text{root}(f) = LNC(F_{n-1}) \text{ and } f \text{ is unifiable with the f-structure of } F_{n-1} \}$$

That is, the set of competing fragments at any step are those whose c-structures have the same root node as $LNC(F_{n-1})$ and whose f-structures are unifiable with the f-structure of $F_{n-1}$. The competition probability remains dependent on each particular step in the process, as in (225):

$$\quad \quad \quad \quad \quad \quad (225) \quad CP(f \mid CS) = CP(f) = \frac{P(f)}{\sum_{f' : \text{root}(f') = \text{root}(f) \text{ and } f' \text{ is unifiable with } F_{n-1}} P(f')}$$

In order to extract valid representations, one needs to sample from the (smaller) set of competing fragments derived via (225) as well as those produced via the conditional probabilities in (220)-(221), and apply Completeness and Coherence post hoc.

3. In addition to the steps outlined thus far, the LFG Coherence check is enforced at each step, ensuring that each grammatical function (SUBJ, OBJ etc.) present in the f-structure is governed (i.e. required
to be present) by a PRED. This means that in model M3, we are dealing only with well-formed c-structures which correspond to coherent and consistent f-structures. That is, structures which satisfy LFG’s Uniqueness check, thereby permitting unification only where exactly appropriate. Bod & Kaplan (op cit.) note that given the non-monotonic property of the Completeness check (i.e., that each grammatical function governed by a PRED is present in the f-structure), namely that its satisfiability depends not only on all steps in the process to date, but also on subsequent ones, stepwise definitions of competition sets (as in (222) and (224)) cannot be given. Completeness, therefore, can only be enforced after all other validity sampling has taken place.

In sum, we note that in models 1-3 the category matching condition is enforced on-line, and all LFG checks are either performed on-line or post hoc, whereas given the non-monotonic nature of the Completeness check, this can only ever be enforced off-line. A number of other models can also be envisaged where various combinations of these conditions are evaluated at different stages in the process. For example, Bod & Kaplan (1999) note the possibility of a Model M0, where the competition set postpones all LFG grammaticality checks in addition to DOP’s category-matching stipulation to the end of the process. In this case the competition set is equal to the set of all fragments in the corpus, with the stochastic process operating in a completely unconstrained manner. All sampling is done off-line and the conditional probabilities given in (220) and (221) are relied upon to account for the well-formedness constraints. An illustration of the different effects of these competition sets is given in section 4.2.

Let us now show an example of the LFG-DOP composition operator with fragments from the parse tree for *John eats pizza* (cf. (25)). In example (226)\(^5\) the SUBJ NP node in the leftmost tree is vacant, and the rightmost NP tree can be substituted for this node:

(226)

\[
\begin{align*}
\text{NP} & \quad \text{VP} \\
\text{V} & \quad \text{NP} \\
\text{eats} & \quad \text{[SUBJ [NUM SG] PRET [VAT [↑ SUBJ [↑ OBJ]]] TENSE PRES OBJ [NUM SG]]} \\
\end{align*}
\]

\[
\text{John} \quad \text{PRED [NUM SG]}
\]

The respective f-structures are then unified to give the \langle c, f \rangle fragment in (227):

\(^5\)Bod & Kaplan (1999) note that if such an example were the only one in the corpus, it would amount to saying that SUBJs and OBJs must agree in number in English. However, some languages, such as the pidgin Central Hiri Motu of Papua New Guinea, do have verbal suffixes that indicate the person and number of the direct object, but no verbal subject affixes (Bresnan, personal communication). Given this possibility, maintaining this sort of fragment is linguistically justified. Nevertheless, we suppose that such effects will be minimized if a suitably large, representative English LFG-DOP treebank is used.
We can now substitute the pizza NP for the OBJ NP node in the tree in (227), and unify the corresponding f-structures to give the resultant well-formed \( \langle c, f \rangle \) fragment in (228):

Throughout the derivation of this \( \langle c, f \rangle \) pair, we have satisfied DOP’s Root condition (leftmost substitution of ‘like’ categories only), as well as the Uniqueness, Completeness and Coherence grammaticality conditions of LFG. As a consequence, the resultant structures in (228) are valid. This amounts to using the third competition set (M3) of Bod & Kaplan (op cit.) as input into our LFG-DOP probability model.

Of course there will be many other possible derivations which contribute to the overall probability of the sentence. Note that if we enforce LFG’s grammaticality checks on-line, leftmost substitution of non-\textit{Discard} fragments reduces the size of the competition set for future iterations of the composition process. In (226), for instance, enforcing the Uniqueness condition on-line (model M2-M3) prevents any fragment other than an NP with \text{NUM=SG} from being substituted into the SUBJ NP slot, as in (227). In Tree-DOP, any NP could be substituted at this node.\footnote{Cormons (1999) notes that leftmost substitution can affect the probability model over other forms of substitution (cf. (265)-(266), p.135). We address this further in section 6.3.5.} For example, we could substitute the tree (229) into the NP slot in the CFG tree in (227) to give a well-formed analysis in Tree-DOP:

However, if we attempt to use LFG-DOP composition, we see that the f-structure fragments which we are attempting to unify in (230) contain incompatible values for the NUM value, rendering unification unsuccessful:

\footnote{Cormons (1999) notes that leftmost substitution can affect the probability model over other forms of substitution (cf. (265)-(266), p.135). We address this further in section 6.3.5.}
Nevertheless, the f-structure fragments in (230) could be unified if the constraint that the OBJ of *eats* is SG were relaxed. This is, of course, nothing to do with a fact about English (cf. note 5, this chapter), so this possibility needs to be maintained as there is no universal principle which rules out such fragments. This amounts to using the second competition set (M2) of Bod & Kaplan (*op cit.*) as input into our LFG-DOP probability model.

Using *Discard*, however, we can remove the **NUM=PL** constraint from the f-structure fragment for *cakes* in (230), as well as the **NUM=SG** constraint from the OBJ slot, and thereby enable unification,\(^7\) with the OBJ of *eats* unspecified for number, as in (231):

These examples show how the unification element of LFG can prevent the Uniqueness condition of LFG from being flouted, as well as how this can be circumvented using *Discard* to enable analyses that would otherwise be ruled out. All such possible derivations are regarded as ungrammatical with respect to the corpus. The other grammaticality constraints of LFG—Completeness and Coherence—can also be used to prevent certain analyses. For example, enforcing the Coherence condition will rule out (232) as being accepted as a valid structure:

\(^7\)We can do the same for the SUBJ value too, to allow an LFG-DOP analysis of the ill-formed string *cakes eats John*, for instance. We discuss these sorts of examples further in section 4.2.
The semantic form for *swim* stipulates that it subcategorizes for a SUBJ only, which leaves the f-structure OBJ stranded, it not being governed by the local predicate. This amounts to using the third competition set \((M3)\) of Bod & Kaplan (*op cit.* ) as input into our LFG-DOP probability model. Similarly, enforcing the Completeness condition will rule out \((233)\), as *devour* is a transitive verb, but there is no OBJ in the f-structure:

\[
(233) \quad \begin{array}{c}
\textbf{S} \\
\textbf{NP} \\
\textbf{VP} \\
\textbf{S} \\
\textbf{NP} \\
\textbf{VP} \\
\end{array}
\quad \begin{array}{c}
\text{John} \\
\text{swims} \\
\text{pizza} \\
\text{John} \\
\text{devours} \\
\end{array}
\]

\[
\begin{array}{c}
\text{SUBJ} \\
\text{PRED} \\
\text{TENSE} \\
\text{OBJ} \\
\end{array}
\quad \begin{array}{c}
\text{PRED} \text{ ‘John’} \\
\text{NUM} \\
\text{PRES} \\
\text{NUM} \\
\end{array}
\quad \begin{array}{c}
\text{PRED} \text{ ‘pizza’} \\
\text{SG} \\
\text{PRED} \text{ ‘devour((↑SUBJ)(↑OBJ))’} \\
\text{SG} \\
\end{array}
\]

As stated earlier, the Completeness condition can only ever be enforced *post hoc*. The only way in which the \((c,f)\) structure pairs in \((230)\), \((232)\) and \((233)\) would be acceptable would be if we were to use the first competition set \((M1)\) of Bod & Kaplan (*op cit.* ) as input into our LFG-DOP probability model.

We have introduced the *Discard* operation of Bod & Kaplan (*op cit.* ) and given some examples of how it can relax certain constraints to allow readings of strings which would otherwise be ruled out. Nevertheless, it should be reiterated that for an analysis to be deemed valid, there must be at least one derivation of that string without *Discard*, i.e. via *Root* and *Frontier* only. Given this, unless there was at least one other derivation possible, the \((c,f)\) analysis derived in \((231)\) would render *John eats cakes* as ungrammatical *with respect to the corpus*. For example, we next show that an analysis of *John swims* is obtainable from fragments derived from *Discard* (see Figure 4.2); if this were the only analysis possible, the sentence would be deemed ungrammatical. However, *John swims* can also be processed by the fragments in Figure 4.1 obtained via *Root* and *Frontier*. This means that as there is at least one analysis possible other than via *Discard*, the sentence is indeed grammatical with respect to the corpus.

We provide in Figure 4.2 the complete set of LFG-DOP \((c,f)\) fragment pairs for *John swims*. Note that for any LFG-DOP model containing *Discard*, these fragments are additional to the original fragment pairs in Figure 4.1. Contrasting this corpus with Figure 4.1, we observe an additional 18 fragments. The only constraints that *Discard* can relax in this corpus are \(\text{NUM=SG}\) and \(\text{TENSE=PRES}\). Where one of these occurs without the other, it is the only constraint that can be relaxed. When both occur together in a fragment in Figure 4.1, each can be relaxed in turn, and they can both be relaxed together, hence the high number of additional \((c,f)\) fragments just for a simple sentence. It is evident that for treebanks of any realistic size,
Figure 4.2: The additional LFG-DOP fragments via Discard for the sentence John swims
the number of generalized fragments produced by Discard is going to be huge. We know already from DOP models that the number of subtree fragments produced is large: for LFG-DOP the number of fragments will be of an order of magnitude greater, unless they can be pruned in an informed manner. We discuss this further in section 4.6.1.

For the further effects of the Discard operation to be felt, we show the probabilities for some simple sentences using non-Discard fragments, as well as probabilities for the same sentences derived from all fragments. As well as the LFG-DOP fragments in Figure 4.1 and Figure 4.2, we assume an LFG-DOP corpus for Peter laughs. Apart from the different lexical items, these are identical to the fragments in Figure 4.1 and Figure 4.2. Given the existence of these corpora, we can now provide representations for the four sentences in (234):

(234) a. John swims.
    b. John laughs.
    c. Peter laughs.
    d. Peter swims.

Assuming that only non-Discard fragments are used, these four sentences have exactly the same probabilities as they did under Tree-DOP (cf. Figure 2.3, p.40 and resultant discussion): John swims and Peter laughs have probabilities \( \frac{7}{14} \), whereas the other two sentences each have a probability of \( \frac{5}{14} \). This disparity is accounted for in LFG-DOP by the presence of the fragments in (235), each of which has a probability of \( \frac{1}{14} \) (or \( \frac{2}{49} \)):

![Diagram of sentence representations](image)

If we now recalculate these probabilities using the full set of 56 fragments (10 fragments each via Root and Frontier for John swims and Peter laughs (cf. Figure 4.1), and 18 fragments for each of these two strings (cf. Figure 4.2) via Discard), we obtain the results in (236):
(236)  a. \( P(\text{John swims}) = 0.0498 \approx 1/20 \)
    b. \( P(\text{John laughs}) = 0.01855 \approx 1/54 \)
    c. \( P(\text{Peter laughs}) = 0.0498 \approx 1/20 \)
    d. \( P(\text{Peter swims}) = 0.01855 \approx 1/54 \)

This assumes that competition set \( M3 \) is input into the probability model. Given that these are the only four sentences derivable from these corpora, it is clear that using the 20 non-\textit{Discard} fragments, the probabilities for these sentences occupy 100% of the probability space (i.e. their probabilities sum to 1). Now we see that just using the well-formed (i.e. non-\textit{Discard}) fragments in (236), the probabilities sum to 0.1367 (\( \approx 1/7 \)) of the total probability space. Consequently, it is clear to see that the introduction of \textit{Discard} has had quite an effect on the probability models. We discuss this further in section 6.3.6, where we report that \textit{Discard} fragments may occupy 95% or more of the probability mass.

Of course, the probabilities in (236) are\textit{ not} the probabilities of the four sentences: we need to add to these numbers the probabilities of the sentences via the set of \textit{Discard} fragments, plus their interaction with the non-\textit{Discard} fragments. As a point of comparison, in (237) we give the probabilities of the four sentences just using the \textit{Discard} fragments out of the set of all fragments:

(237)  a. \( P(\text{John swims}) = 0.1465 \approx 1/7 \)
    b. \( P(\text{John laughs}) = 0.054 \approx 1/18 \)
    c. \( P(\text{Peter laughs}) = 0.1465 \approx 1/7 \)
    d. \( P(\text{Peter swims}) = 0.054 \approx 1/18 \)

Comparing these figures with those in (236), we see that the probabilities via \textit{Discard} fragments are about three times larger than for non-\textit{Discard} fragments. We note that adding the probabilities in (236) to those (237) does not sum to 1. The probabilities in (236) are calculated by using just the well-formed fragments, and those in (237) are formed via the \textit{Discard}-generated fragments only. Summing both sets of probabilities, we see that these probabilities occupy 53.35% of the available probability space. The remaining 46.65% is taken up by those derivations formed by combining grammatical with ungrammatical fragments (cf. (240) overleaf). The final probabilities for the four sentences are given in (238):

(238)  a. \( P(\text{John swims}) = 0.352 \approx 1/3 \)
    b. \( P(\text{John laughs}) = 0.148 \approx 1/7 \)
    c. \( P(\text{Peter laughs}) = 0.352 \approx 1/3 \)
    d. \( P(\text{Peter swims}) = 0.148 \approx 1/7 \)

As these four strings are the only possible sentences given the treebank, the probabilities sum to 1 as required. Comparing these figures with the original probabilities just using non-\textit{Discard} fragments, we see that the ratio of \( P(\text{John swims}) : P(\text{Peter swims}) \) was 1.4, but using \textit{Discard} it is 2.37. There is no doubt that the introduction of \textit{Discard} alters the LFG-DOP probability models in an undesirable way (cf. section 6.3.6 for attempts to redress the balance in favour of the non-\textit{Discard} fragments).
It should also be noted that given Bod & Kaplan’s definition of grammaticality, no derivations produced solely via *Discard* fragments result in well-formed structures. For example, we can process *John swims* via the LFG-DOP fragments in (239), which are taken from Figure 4.2, but the resultant f-structure is missing NUM=SG:

(239)  
\[
\begin{array}{c}
\text{NP} \\
\text{VP} \\
\text{S} \\
\end{array}
\quad
\begin{array}{c}
\text{SUBJ} \\
\text{PRED} \quad \text{‘John’} \\
\end{array}
\quad
\begin{array}{c}
\text{V} \\
\text{swims} \\
\end{array}
\]

By definition, each fragment derived via *Discard* is missing at least one feature, so when two such fragments are combined these missing features will be perpetuated still further. It is obvious that combinations of non-*Discard* fragments will result in well-formed structures. While we would not wish to advocate a change in the definition of grammaticality, it is worthwhile making the observation that certain combinations of non-*Discard* fragments with *Discard* fragments may result in well-formed structures, as in (240):

(240)  
\[
\begin{array}{c}
\text{NP} \\
\text{VP} \\
\text{S} \\
\end{array}
\quad
\begin{array}{c}
\text{SUBJ} \\
\text{PRED} \quad \text{‘John’} \\
\text{NUM} \quad \text{SG} \\
\end{array}
\quad
\begin{array}{c}
\text{V} \\
\text{swims} \\
\end{array}
\]

The leftmost \(\langle c, f \rangle\) fragment comes from Figure 4.1, and the rightmost fragment comes from Figure 4.2. The rightmost tree can be substituted at the VP node in the leftmost tree, producing a well-formed Tree-DOP structure, and the two f-structures can be successfully unified to produce a well-formed f-structure with no missing features.

**Observations on Corpus Size**

It is evident that the number of fragments will grow with the length of the input, so the *Discard* operation will need to be constrained considerably if LFG-DOP is to prove a practical environment for NLP applications. We discuss ways of reducing the scope of *Discard* in sections 4.4 and 4.5. To give some idea of the growth of fragments, let us again take *John swims* as an example. From Figure 4.1, it should be clear that for LFG-DOP models with no *Discard*, the number of fragments will always be as in Table 4.1.

<table>
<thead>
<tr>
<th>Fragments</th>
<th></th>
<th>#NP</th>
<th>#VP</th>
<th>#V</th>
<th>#Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>#S</td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 4.1: Number of LFG-DOP Fragments for *John swims* (no *Discard*).

If we increase the number of NP and S-features permitted for *John swims*, the increase in the number of fragments produced is shown in Table 4.2. It is not unreasonable to posit the existence of this number of combinations of features. For NP we could argue for **NUMBER, GENDER, PERSON, CASE** (although **CASE**
Table 4.2: The Effect of Discard on the Number of Fragments produced

<table>
<thead>
<tr>
<th>Features</th>
<th>Fragments</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>#NP</td>
<td>#S</td>
<td>#NP</td>
<td>#VP</td>
<td>#V</td>
<td>Total</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>16</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>28</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>32</td>
<td>4</td>
<td>12</td>
<td>8</td>
<td>56</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>64</td>
<td>8</td>
<td>24</td>
<td>16</td>
<td>112</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>128</td>
<td>16</td>
<td>48</td>
<td>32</td>
<td>224</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>24</td>
<td>2</td>
<td>10</td>
<td>8</td>
<td>44</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>46</td>
<td>2</td>
<td>21</td>
<td>19</td>
<td>88</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>68</td>
<td>2</td>
<td>32</td>
<td>30</td>
<td>132</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>44</td>
<td>4</td>
<td>18</td>
<td>14</td>
<td>80</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>90</td>
<td>8</td>
<td>37</td>
<td>29</td>
<td>164</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>184</td>
<td>19</td>
<td>76</td>
<td>60</td>
<td>336</td>
</tr>
</tbody>
</table>

may be considered redundant given the presence of LFG’s grammatical functions of SUBJ, OBJ etc), and at
the sentential level we might reasonably have TENSE, NEGATION, ASPECT, PASSIVE, for example. Recall
that for the small-scale experiments involving the sentences in (234), where each f-structure contained just
one NP feature and one S feature, we noted an increase from 20 fragments prior to the Discard operation to
56 following its application. In calculating the probabilities for these sentences, we noted an increase in the
ratio of \( P(\text{John swims}) \):\( P(\text{Peter swims}) \) from 1.4 without Discard, to 2.37 with Discard. If we choose instead
to have 4 NP features and 2 S features, we can expect to see a further ‘downgrading’ of the probabilities
of the ‘well-formed’ (non-Discard) fragments, in favour of the fragments produced via Discard. A further,
obvious corollary is an increase in processing time for the same sentences given the number of fragments
from which to sample.

For the most part, the increases in Table 4.2 are linear. For instance, the rate of growth in S-fragments in
the first part of the table is \( 2^{\#\text{features}+2} \), as we have to allow the possibility that either the NP or S-features
(or both) will be absent. Consequently the number of S-fragments where we allow 2 NP features and 1
sentential feature is \( 2^5 \), as there are in fact three combinations of permissible NP-features (2, 1 or 0), and
two combinations of S-features (1 or 0). Some, such as NP, rise at a rate of \( 2^{\#\text{NP-features}} \) for all models, as
the number of possible combinations of features within an NP f-structure is independent of the number of
S-features.

It should be clear that an exponential rate of increase will occur with transitives (and verbs of greater
valency), as the number of possible permutations using Discard to remove features from both SUBJ and
OBJ NPs will multiply rapidly. As Cormons (1999) shows with the simple example (241), there are a
potential \( 256 \) (\( c,\phi,\delta \)) derivable fragments just for the VP:

126
Presumably Comons calculates this by assuming that for each of the four possible tree-structures in (242) derivable from the VP in (241), there are $2^6$ possible combinations of the f-structure in (241). That is, the number of fragments is $2^{n+1}$ in the general case where $n$ is the number of features. This allows for the possibility that all features may be absent. Comons (1999:77) states the number of fragments to be $2^n$, but we presume that he omitted to consider this latter possibility. $4 \times 2^6$ is equal to $2^8$, or 256:

These can be restricted to 16 by linking words in the c-structure with their corresponding f-structure predicates at the time of fragmentation. Notwithstanding this, it can be seen that the amount of data to be dealt with is non-trivial. We give Comons’ results (op cit., p.77) from experimenting on a corpus of 440 sentences in Table 4.3, where the number of c-structures ($n_c$) and f-structures ($n_f$) derivable by limiting the depth ($D_{max}$) are given.

<table>
<thead>
<tr>
<th>$D_{max}$</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n_c$</td>
<td>18960</td>
<td>41744</td>
<td>86770</td>
</tr>
<tr>
<td>$n_f$</td>
<td>325 025 676</td>
<td>46 304 773 282</td>
<td>11 934 542 233 764</td>
</tr>
</tbody>
</table>

Table 4.3: The Explosion of LFG-DOP fragments in a corpus of 440 sentences

Comons concludes (ibid) that “it is unthinkable to explicitly create all fragments”. He proposes to cut down the number of fragments using simple heuristics:

- limiting the depth of c-structures to 3;
- for each $\langle c, \phi, f \rangle$ fragment, each sub-f-structure of $f$ which is not the image of a node in $c$, must only contain atomic attributes;
- considering that for each attribute having an empty f-structure as its value, this attribute has no meaning and can be deleted.
An example of the latter restriction is shown in (243):

(243) \[
\begin{array}{c}
\text{VP} \\
\text{swims}
\end{array}
\begin{array}{c}
\text{SUBJ} \\
[]
\end{array}
\begin{array}{c}
\text{PRED} \\
\text{swim}(\text{SUBJ})'
\end{array}
\begin{array}{c}
\text{TENSE} \\
\text{PRES}
\end{array}
\] 

\[
\Rightarrow
\begin{array}{c}
\text{VP} \\
\text{swims}
\end{array}
\begin{array}{c}
\text{PRED} \\
\text{swim}(\text{SUBJ})'
\end{array}
\begin{array}{c}
\text{TENSE} \\
\text{PRES}
\end{array}
\]

Here the leftmost fragment pair is reducible to the rightmost fragment pair. While this reduction of fragments makes Cormons’ experiments (more) practicable, one wonders whether there are any wider repercussions of this adoption. For instance, in the LFG rule (244), the second annotation on the VP indicates that subject NPs occur to the immediate left of tensed VPs, \textit{whatever} the value of the TENSE attribute may be:

(244) \[
S \rightarrow \begin{array}{c}
\text{NP} \\
(\text{SUBJ}) = \downarrow
\end{array}
\begin{array}{c}
\text{VP} \\
\uparrow = \downarrow
\end{array}
(\text{TENSE})
\]

Given that the value is omitted here, partial fragments will be constructed with a TENSE attribute with no value. One has to presume that Cormons’ deletion operation will then remove TENSE from these fragments, thereby allowing subjects to co-occur with untensed VPs.

Furthermore, Cormons proposes choosing a compact method of representing fragments. He uses indexed trees to represent LFG structures, so that the structures in (241) are represented as in (245):

(245) \[
\begin{array}{c}
(S.1 \text{ (NP.2 John.2)}) \\
(VP.1 \text{ (V.1 likes.1)})
\end{array}
\begin{array}{c}
\text{NP.3 Mary.3))}
\end{array}
\]

The corresponding hash-table in (246) contains:

\[
1 \rightarrow \begin{array}{l}
\text{[ (PRED = like (SUBJ,OBJ))} \\
\text{(SUBJ = 2)} \\
\text{(OBJ) = 3])}
\end{array}
\]

\[
2 \rightarrow \begin{array}{l}
\text{[ (PRED = John)} \\
\text{(NUM = SG)} \\
\text{(GEND = MASC)]}
\end{array}
\]

\[
3 \rightarrow \begin{array}{l}
\text{[ (PRED = Mary)} \\
\text{(NUM = SG)} \\
\text{(GEND = FEM)} \\
\text{(CASE = ACC)]}
\end{array}
\]

Subtrees are handled in the same fashion, as in (247):
Cormons then associates to each indexed subtree a list indicating valid fragments which can be constructed from it, as in (248):

\[
\begin{pmatrix}
1 & -1 & 1 & 2 \\
2 & 1 & 2 & 2
\end{pmatrix}
\]

Here 1 indicates that the attribute must be present, -1 absent, and 0 indicates that either is possible. We revisit these, as well as other ways of limiting the scope of *Discard* in section 4.4.

### 4.1.2 A New LFG-DOP Model

Bod & Kaplan (1999) develop a new LFG-DOP model which differs from the model of Bod & Kaplan (1998) only in the decomposition operation. In describing the original model above, we noted that as well as *Root* and *Frontier* operating as in Tree-DOP, the notion of \(\phi\)-accessibility was needed to denote which parts of the f-structure should be linked to c-structure fragments. We repeat here Bod & Kaplan’s (1998) definition of \(\phi\)-accessibility:

“An f-structure unit \(f\) is \(\phi\)-accessible from a node \(n\) iff either \(n\) is \(\phi\)-linked to \(f\) (that is, \(f = \phi(n)\)) or \(f\) is contained within \(\phi(n)\) (that is, there is a chain of attributes that leads from \(\phi(n)\) to \(f\)).

\(\phi\)-accessibility is replaced in the new LFG-DOP model by the notion of *support* (Bod & Kaplan, 1999:14, original example (36)):

“Intuitively, a node carries information about, or supports, its corresponding f-structure unit and the function and features that are closely connected to it. This notion is made precise in the following recursive definition:

\[
\begin{align*}
\text{(249)} & \quad \text{A node } n \text{ supports an f-structure unit } u \text{ if and only if} \\
\text{(a)} & \quad u = \phi(n), \text{ or} \\
\text{(b)} & \quad u \text{ is the value of some grammatical-function attribute in } \phi(n), \text{ or} \\
\text{(c)} & \quad u \text{ is the value of some non-grammatical-function attribute in an f-structure } v, \text{ where} \\
\end{align*}
\]

That is, in the decomposition operations of the two models, the two somewhat looser definitions of \(\phi\)-accessibility are made explicit by the three definitions of support in (249). Nevertheless, both models allow the same set of fragments to be produced. The new notion of support is perhaps best explained by means of an example. Let us add node identifiers and explicit \(\phi\)-links to (228), p.119, to produce (250):
Here, the SUBJ NP node supports only the attribute-value pair \texttt{NUM=SG}. The PRED value is not supported by the NP as it is \( \phi \)-linked to the lexical item \textit{John}, which violates the third condition above. The John:3 node, therefore, is the only item which can support the PRED value (PRED \( \phi \) ‘John’ in (250)). Bod & Kaplan (1999:14) state that this definition “maintains the fundamental two-way connection between words and meanings, in that semantic forms are supported only by the lexical items they correspond to”. Of course, as Cormons shows (cf. (241) above), such a definition will cut down considerably the number of fragments produced.

In order to illustrate the effect of condition (249b), we need a more complex example, such as (251):

Here the root S node supports the grammatical functions and features of the outermost f-structure, as usual (via 249a), but also those features \textit{inside} the top-level functions, as in (252):

The grammatical functions of the COMP are not supported by the root, however.

Bod & Kaplan (1999:15) continue: “In the obvious way we extend the definition of support from individual nodes to the collection of nodes that make up a subtree. We say that a subtree supports the (sub) f-structure consisting of the union of all the f-structure units supported by its nodes”. If we extract the VP subtree from (250), together with the f-structure units supported by its nodes, fragment (253) is derived:
Here we observe that the VP subtree supports everything in (250) except the PRED of the SUBJ function, which is supported by the lexical item itself. The NUM feature of the SUBJ and its value remain despite the absence of the NP node, as this A-V pair “is an unlinked feature of an immediate function of the VP node’s f-structure” (ibid.), i.e. condition (249c).

The initial set of fragments of this new LFG-DOP model can now be specified using DOP’s Root and Frontier operations together with the notion of f-structure support in (249), as in (254):

(254) “Given an LFG representation \(\langle c, \phi, f \rangle\), the triple \(\langle c', \phi', f' \rangle\) is an LFG-DOP fragment if \(c'\) is a subtree produced from \(c\) by the Tree-DOP decomposition operations, \(f'\) is the substructure of \(f\) supported by the nodes in \(c'\) and \(\phi'\) is the sub-correspondence of \(\phi\) that relates the nodes in \(c'\) to the units in \(f'\)” (Bod & Kaplan, 1999:16).

### 4.1.3 Alignment Problems in LFG-DOP

Both LFG-DOP models presuppose the ability to link c-structure nodes with their corresponding f-structure attributes. While this has been unproblematic for the examples shown to date, there are some rather more difficult instances for which some solution needs to be found. We consider such examples to be outside the scope of the general aims of this thesis, but present them here as worthy of future consideration. For instance, one can have discontinuous elements in the c-structure which need to ‘come together’ in the f-structure, as in (255):

(255) ![Diagram of S, NP, VP, SUBJ, PRED, OBJ with John, phoned, Mary, up]

In addition, there are examples of elements in the f-structure which have no overt c-structure nodes to which they can be linked, such as in (256):

(256) ![Diagram of SUBJ, PRED 'John', OBJ, PRED 'Mary']
This latter example merits some discussion. The LFG device of functional control makes explicit the 'understood' SUBJ of *go* via an equation in the lexical entry for *try* († SUBJ) = († XCOMP SUBJ), illustrating that *John* is SUBJ both of *try* and *go*.\(^8\) Nevertheless, we would be wrong in physically linking the *John* c-structure node to both SUBJ slots in the f-structure, as the line linking these two SUBJ slots indicates re-entrance of the structure, namely that there are two distinct paths to this structure, which are *token* (and by implication *type*) identical, which is different from having two copies of the same structure in different locations, which may be *type*, but not *necessarily token* identical.

### 4.2 DOP adds Robustness to LFG

LFG-DOP (Bod & Kaplan, 1998) adds a measure of robustness (both with respect to unseen as well as ill-formed input) not available to models based solely on LFG. For example, Bod & Kaplan (op cit.) assume the treebank (cf. Figure 4.3 below) for the sentences in (257):

(257) a. People walked.

    b. John fell.

Given such a treebank, LFG-DOP models can be constructed where, for the two new strings in (258), the unmarked interpretation is less likely than the two specific interpretations:

(258) a. John walked.

    b. People fell.

Furthermore, the intuitively correct ones are selected for each corresponding verb. Their example LFG-DOP treebank, shown here in Figure 4.3, is simplified somewhat to cut down on the number of fragments obtained and thereby to facilitate the calculations required. In Figure 4.3, the c-structures map to their respective f-structures firstly without *Discard* (i.e. via *Root* and *Frontier* only), and secondly with *Discard*. The LFG-DOP treebank for *People walked* will, apart from different leaves and PRED values, differ from Figure 4.3 only with respect to the value of the NUM attribute. For *John fell*, therefore, there are 12 fragments with either NUM=SG or unmarked, and correspondingly for *People walked* there are 12 fragments with either NUM=PL or unmarked. The \(S(N_{P,V_P})\) c-structure fragment with NUM=unmarked is the only fragment

Figure 4.3: The LFG-DOP treebank for the sentence John fell
produced twice, once from either sentence. Each other fragment occurs once only, with the total number of fragments for each category being S:16, NP:4, and VP:4.

Let us now derive the probabilities for the new sentences in (258) with respect to the probability models M1 and M2 above. Bod & Kaplan (op cit.) employ an alternative (but equivalent) notation for the derivations of sentences. One such derivation of John walked is given in (259):

(259) \[ S/NP-VP/U \circ NP/John/SG \circ VP/walked/U \]

(259) is equivalent to the derivation in (260):

(260)

\[
\begin{align*}
& \text{S} \quad \text{NP} \quad \text{VP} \\
& \quad \text{SUBJ} \\
& \quad \text{PRED} \quad \text{‘walk’(\text{SUBJ})} \\
& \quad \text{walked} \\
& \quad \text{PRED} \quad \text{‘walk’(\text{SUBJ})} \\
& \quad \text{PRED} \quad \text{‘John’} \\
& \quad \text{NUM} \\
& \text{NP} \\
& \text{John} \\
& \text{VP} \\
& \text{PRED} \quad \text{‘walk’(\text{SUBJ})} \\
& \text{SUBJ} \\
\end{align*}
\]

Model M1 checks only for the Tree-DOP root-matching condition, and the competition sets are fixed independent of the derivation. Therefore, the probability of the derivation in (259) (or (260)) is 2/16 x 1/4 x 1/4 = 2/256 (cf. (261a)). There are 5 different derivations for John walked with NUM=SG, as in (261):

(261) a. \[ S/NP-VP/U \circ NP/John/SG \circ VP/walked/U = 2/16 \times 1/4 \times 1/4 = 2/256 \]

b. \[ S/NP-VP/SG \circ NP/John/SG \circ VP/walked/U = 1/16 \times 1/4 \times 1/4 = 1/256 \]

c. \[ S/NP-VP/SG \circ NP/John/U \circ VP/walked/U = 1/16 \times 1/4 \times 1/4 = 1/256 \]

d. \[ S/John-VP/SG \circ NP/John/SG \circ VP/walked/U = 1/16 \times 1/4 = 1/64 \text{ (or 4/256)} \]

e. \[ S/NP-walked/U \circ NP/John/SG = 1/16 \times 1/4 = 1/64 \text{ (or 4/256)} \]

Summing the probabilities of the respective derivations enables the calculation of the probability of the sentence: John walked: NUM=SG = 12/256 = 0.046875. However, John walked can also be analysed with NUM=PL, as in (262):

(262) a. \[ S/NP-VP/U \circ NP/John/U \circ VP/walked/PL = 2/16 \times 1/4 \times 1/4 = 2/256 \]

b. \[ S/NP-VP/PL \circ NP/John/U \circ VP/walked/PL = 1/16 \times 1/4 \times 1/4 = 1/256 \]

c. \[ S/NP-VP/PL \circ NP/John/U \circ VP/walked/PL = 1/16 \times 1/4 \times 1/4 = 1/256 \]

d. \[ S/John-VP/U \circ VP/walked/PL = 1/16 \times 1/4 = 1/64 \text{ (or 4/256)} \]

e. \[ S/NP-walked/PL \circ NP/John/U = 1/16 \times 1/4 = 1/64 \text{ (or 4/256)} \]

This gives \[ P(John \text{ walked}: \text{NUM}=\text{PL}) = 12/256 = 0.046875 \]. The analysis where NUM is unmarked has the derivations in (263):
(263) a. $S/\text{NP-VP/U} \circ \text{NP/John/U} \circ \text{VP/walked/U} = 2/16 \times 1/4 \times 1/4 = 2/256$
   
   b. $S/\text{John-VP/U} \circ \text{VP/walked/U} = 1/16 \times 1/4 = 1/64$ (or $4/256$)
   
   c. $S/\text{NP-walked/U} \circ \text{NP/John/U} = 1/16 \times 1/4 = 1/64$ (or $4/256$)

Therefore, $P(\text{John walked:NUM=U}) = 10/256 = 0.039$.

Now given (221), p.116, we can calculate the probabilities of each of these analyses given all the analyses of $\text{John walked}$, as in (264):\(^9\)

(264) a. $P(\text{NUM=SG} \text{valid and yield } = \text{John walked}) = 12/34 = 0.353$
   
   b. $P(\text{NUM=PL} \text{valid and yield } = \text{John walked}) = 12/34 = 0.353$
   
   c. $P(\text{NUM=U} \text{valid and yield } = \text{John walked}) = 10/34 = 0.294$

It is clear that the two specific representations are equally probable and more likely than the unmarked representation.

Calculating the probabilities of the same representations using model M2 will demonstrate the impact of using different competition sets. As well as enforcing Tree-DOP's root-matching condition, the requirement that only consistent analyses are derived needs to be enforced. As soon as a fragment instantiates the NUM feature, the number of fragments from which to sample in the competition set is immediately restricted to fragments of the same NUM value. Therefore, if we choose the NP/John/SG in (259), only three VPs (rather than four in M1) remain in the competition set for the next stage of sampling: VP/fell/SG, VP/fell/U and VP/walked/U. VP/walked/PL is excluded from the probability model as it cannot be combined with NP/John/SG without resulting in an inconsistent derivation. Hence the probability of (259) under M2 is $\frac{4}{3}$ higher than under M1: $2/16 \times 1/4 \times 1/3 = 2/192$.

Of course, the number of possible derivations of $\text{John walked}$ under model M2 remains the same; it is just the probabilities which change slightly. Recalculating the probabilities of the derivations in (261)-(263) gives the new results in (265):

(265) a. $P(\text{NUM=SG and yield } = \text{John walked}) = 35/576 = 0.061$
   
   b. $P(\text{NUM=PL and yield } = \text{John walked}) = 33.5/576 = 0.058$
   
   c. $P(\text{NUM=U and yield } = \text{John walked}) = 22.5/576 = 0.039$

The conditional probabilities for the valid representations are given in (266):

(266) a. $P(\text{NUM=SG | valid and yield } = \text{John walked}) = 70/182 = 0.38$
   
   b. $P(\text{NUM=PL | valid and yield } = \text{John walked}) = 67/182 = 0.37$
   
   c. $P(\text{NUM=U | valid and yield } = \text{John walked}) = 45/182 = 0.25$

These results are biased somewhat by the LFG-DOP stipulation that derivations be composed by leftmost substitution. For the sentence $\text{John walked}$, we typically instantiate the SUBJ NP node by $\text{John:SG}$, which

---

\(^9\)Exactly the same conditional probabilities result from dividing each of the probabilities of $\text{John walked}$ by the sum of the probabilities of all the possible readings of $\text{John walked}$, e.g. $P(\text{NUM=SG | valid and yield } = \text{John walked}) = 0.046875/0.1328125 = 0.353$, *ceteris paribus*.  

135
restricts candidates for future sampling steps. If we were instead to investigate the probabilities of the new string *people fell*, we would get a similar bias for NUM=PL. Note that this may prove useful in practice (if these biases surpass some level of significance): if we already have a plural subject NP, and we come across a verb we do not recognize, at least we prefer it to be plural over the unspecified or singular alternatives, and vice versa. The impact of leftmost substitution on the probability models is discussed further in section 6.3.5.

Bod & Kaplan (1998) observe that each of the LFG-DOP probability models M1–M3 prefer specific over unmarked readings. This is partly due to the fact that more specific readings have more derivations than unmarked analyses, so tend, therefore, to be assigned higher probabilities. Cormons (1999) adds that this preference depends also on the number of feature values present: the more there are, in order to obtain a preference for the more specific readings, the minimal derivation length must be longer.

The bias in favour of more specific analyses can also be seen when LFG-DOP attempts to interpret ill-formed strings, as shown in (267):

\[(267) \]
\[a. \quad * \text{John walk} \]
\[b. \quad * \text{This boy walk} \]

With no other knowledge except that *John, this and boy are [SG], and walk is [PL], three readings—SG, PL and undefined—are formed for (267a), as in (268):

\[(268) \]
\[a. \quad \text{John [SG] \circ walk [\_] = John walk [SG]} \]
\[b. \quad \text{John [\_] \circ walk [PL] = John walk [PL]} \]
\[c. \quad \text{John [\_] \circ walk [\_] = John walk [\_]} \]

That is, all three interpretations are equally probable. For (267b), there are five possible derivations—3 SG, 1 PL, and 1 undefined—as in (269):

\[(269) \]
\[a. \quad \text{this [SG] \circ boy [SG] \circ walk [\_] = this boy walk [SG]} \]
\[b. \quad \text{this [SG] \circ boy [\_] \circ walk [\_] = this boy walk [SG]} \]
\[c. \quad \text{this [\_] \circ boy [SG] \circ walk [\_] = this boy walk [SG]} \]
\[d. \quad \text{this [\_] \circ boy [\_] \circ walk [PL] = this boy walk [PL]} \]
\[e. \quad \text{this [\_] \circ boy [\_] \circ walk [\_] = this boy walk [\_]} \]

This shows a clear bias in favour of the effects of subject-verb agreement for English consistent with the weight of (syntactic) evidence. This is observed for all LFG-DOP probability models.

### 4.2.1 Other Methods of Disambiguating LFG Analyses

A number of other proposals have been made for filtering and ranking parsing ambiguities in LFG. We shall describe the details of each in turn, and comment on their merits compared to LFG-DOP.
Ranking LFG Analyses

Johnson (1996) begins by explaining that the well-understood techniques for establishing preferences for parses in probabilistic FSGs and CFGs do not extend to unification-based grammars because of the dependencies that exist in the latter’s ‘rules’ (cf. Abney (1997), and section 6.3.5 below). He exemplifies this claim with respect to the small grammar in (270):

\[
S \rightarrow \text{NP} \quad \text{VP}
\]
\[
\text{(} \uparrow \text{SUBJ}) = \downarrow \quad \uparrow = \downarrow
\]
\[
\text{VP} \rightarrow \text{V} \quad \text{NP}
\]
\[
\uparrow = \downarrow \quad \text{(} \uparrow \text{OBJ}) = \downarrow
\]

(270)

<table>
<thead>
<tr>
<th>Kim</th>
<th>NP</th>
<th>((\uparrow) NUM) = SG</th>
</tr>
</thead>
<tbody>
<tr>
<td>people</td>
<td>NP</td>
<td>((\uparrow) NUM) = PL</td>
</tr>
<tr>
<td>likes</td>
<td>V</td>
<td>((\uparrow) SUBJ NUM) = SG</td>
</tr>
<tr>
<td>like</td>
<td>V</td>
<td>((\uparrow) SUBJ NUM) = PL</td>
</tr>
</tbody>
</table>

Johnson supposes that each lexical item is chosen 50% of the time, so \(P(\text{People like Kim}) = \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{8}\). If feature constraints are ignored (analogous to LFG-DOP probability models M0 and M1), then the grammar in (270) can produce 8 different strings: there are 2 NP subjects, 2 verbs, and 2 different NP objects, therefore \(2^3\). Given that the probability of each string is \(\frac{1}{8}\), the requirement that the probability distribution sum to 1 is maintained. However, incorporating the feature constraints rules out the strings in (271):

\[
(271) \quad \text{a. *Kim like people.}
\]
\[
\text{b. *People likes Kim.}
\]
\[
\text{c. *Kim like Kim.}
\]
\[
\text{d. *People likes people.}
\]

We observe that the \emph{bona fide} strings actually account for only 50% of the probability mass,\(^{10}\) violating the maxim that the probabilities must sum to 1. Hence there is a ‘probability deficit’ caused by feature clashes. The distribution needs to be renormalized to obtain probabilities for the well-formed strings that sum to 1. Johnson goes on to describe a method for approximating a probability distribution over \((c, f)\) pairs of structure. He wants to compute \(P(c, f)\), the joint probability of c-structure \(c\) and f-structure \(f\). This is factored into \(P(f)\), the probability of the f-structure, and \(P(c \mid f)\), the probability of the c-structure given the f-structure, as in (272):

\[^{10}\text{Johnson notes that only strings (271a) and (271b) are ruled out by the invocation of the agreement constraint, but of course the other two strings in (271) also fail to agree, and are ill-formed. This is merely an oversight on Johnson’s part, but it does affect the probabilities of the sentences in his discussion. The correct figures have been inserted here.}\]
\[(272) \quad P(c, f) = P(f)P(c \mid f)\]

\(P(f)\) signifies the probability that \(f\) is the root f-structure of a string. Johnson defines this probability recursively in terms of LFG PRED to PRED dependencies, as in (273):

\[(273) \quad P(f) = P(f \text{ PRED})P(f \mid (f \text{ PRED}))\]

Here \(P(f \mid (f \text{ PRED}))\) represents the probability of the rest of the f-structure given its PRED value. If \(a_1\ldots a_n\) is the list of attributes compatible with \((f \text{ PRED})\), then \(P(f \mid (f \text{ PRED}))\) can be approximated as in (274):

\[(274) \quad P(f \mid (f \text{ PRED})) = \prod_i P((f \ a_i) \mid (f \text{ PRED}), (f \ a_1)\ldots(f \ a_{i-1})) \approx \prod_i P((f \ a_i) \mid (f \text{ PRED}))\]

The approximation in (274) amounts to saying that the choice of an argument or adjunct depends solely on the PRED’s value, and not on the value of any other features. This is clearly incorrect, in that the distinction between sentential and verbphrase adverbial modifiers disappears under this categorization, for example. Nevertheless, as a simple model it may suffice.

Now Johnson turns to the estimation of \(P((f \ a_i) \mid (f \text{ PRED}))\). Given the formulation of LFG, the attributes \(a_i\) may range over a finite set of values, over values with PRED attributes, or over set-valued attributes. Johnson notes that his approximations for each of these possibilities do not account for re-entrancy, but rather copy the re-entrant f-structures en bloc. We discussed earlier the merits of this approach with respect to (256), p.131 (cf. also (285)-(286) below). One problem particular to Johnson’s approach is to ensure that ‘shared’ f-structures such as these are counted only once, otherwise probabilities would be underestimated owing to the probability of the shared argument being multiplied in more than once.

As for the term \(P(c \mid f)\), Johnson assumes that there is only one well-formed c-structure corresponding to a well-formed f-structure, so that \(P(c \mid f) = 1\) when the \((c, f)\) pair is well-formed, and 0 otherwise. Again, this assumption is incorrect. Nevertheless, Johnson notes that reasonable rankings may ensue despite the number of approximations in his model. He observes, however, that estimating the PRED to PRED dependencies will be intractable using “reasonable amounts of data”, and thus proposes to collapse classes of heads with similar distributions together, using a resource like Wordnet (Miller, 1998), or by inducing these from corpora. Nevertheless, he fears that resolving adjunct and conjunction ambiguities may require data tagged, for instance, with predicate-argument structures (cf. section 4.7).

**Optimality Theory Constraint Ranking**

Frank *et al.* (1998) propose an extension of the LFG projection architecture which incorporates ideas from Optimality Theory (OT: Bresnan, 1996). A new projection, *\(o\)-structure*, is posited which determines a preference ranking on a set of sentence analyses. Given that \(o\)-structure is overlaid on the existing grammar, no elements of LFG theory need to be altered.
The rules and constraints used in the derivation of each analysis are recorded in \( \alpha \)-structure, which consists of a set of constants ("optimality marks") projected from \( \theta \)-structure. These optimality marks are introduced by \( \alpha \)-descriptions in the grammar where a disjunction appears, as in (275):

\[
\text{(275)} \quad \text{VP} \rightarrow V \left( \begin{array}{c}
\text{NP} \\
(\uparrow \text{OBJ}) = \downarrow
\end{array} \right) \quad \begin{cases}
\text{PP}^* \\
(\uparrow \text{OBL}) = \downarrow \\
\text{MARK1} \in \alpha^* \\
\downarrow \in (\uparrow \text{ADJUNCTS}) \\
\text{MARK2} \in \alpha^*
\end{cases}
\]

The optimality marks are ordered, marking the constraints as positive and negative, as in (276):

\[
\text{(276)} \quad \text{OPTIMALITY RANKING} \quad \text{MARK1} \quad \text{MARK2} \quad \text{NEUTRAL} \quad \text{MARK3} \quad \text{MARK4}.
\]

MARK1 is the most positive and MARK4 the most negative. Unless explicitly marked, all constraints are adjudged to be NEUTRAL. From the set of all candidate structures, the winner is the one containing the fewest number of the most negative mark. If no winner emerges, the ranking proceeds to the next most negative mark. If there is still no winner, then the positive marks are used to rank the candidate structures.

Consider the two possible structures in (277) for the string *John waited for Mary*, built partially using the rule in (275):

Assuming that the disjunction in the PP element of rule (275) is the only source of optimality marks for
this string, then the uppermost \((c, f)\) pair wins out over the bottommost pair in (277) as it contains an instance of the higher positive mark MARK1, while the bottom structures contain a lower positive mark MARK2. Of course, if we invoke LFG’s grammaticality constraints then structures of both types in (277) will be maintained, depending on the subcategorization requirements of the PRED values. A verb like wait has at least the two semantic forms in (277). However, a verb such as sleep which cannot take an OBL function will occur in structures like the lower ones in (277), as LFG’s coherence check will rule out analyses like the upper structures for sleep, so the lower ones will be maintained and will, trivially, win.

Optimality-theoretic LFG

Bresnan (1998) describes a version of LFG which uses OT constraint satisfaction to identify well-formed linguistic structures. In particular, she shows that re-ranking constraints changes the set of optimal surface forms and uses this to account for a range of dialectical variation in the use of auxiliaries in English.

Johnson (1998a) concentrates on some of the broader implications in going from the standard LFG account (Kaplan & Bresnan, 1982) to OT-LFG. In particular, Johnson notes that OT-LFG requires a fairly radical change to the standard account which may result in the parsing problem for OT-LFG being undecidable. He suggests that just as people have speculated that there is a ‘deep’ relationship between OT and connectionism (Prince & Smolensky, 1998), there is also a close connection between OT-LFG parsing and maximum likelihood parsing. The particular class of probabilistic parsing with which Johnson draws analogies concerns random-field (RF) models. We describe the application of RF-models to constraint-based parsing in section 6.3.5 (cf. Abney, 1997).

In sum, Johnson (1998a) concludes that Bresnan’s (1998) account of OT-LFG suggests novel ways of approaching the parsing and learning tasks of LFG, as LFG is now recast in terms of a series of ranked constraints. He also notes that Bresnan’s analysis could be reformulated in a variety of OT-style frameworks: more important than their representation is the identification of constraint violations, which seems to be a general property of OT accounts.

Comparing these Approaches

We consider LFG-DOP to have a number of advantages over the other approaches presented here. It is true that the original exposition of LFG (Kaplan & Bresnan, 1982) contained no facility whereby different f-structures for the same string could be ranked. The alternative approaches are a step in this direction, and could conceivably be adapted for use in disambiguating translation candidates as well as in parsing. As he admits himself, Johnson’s proposal is rather partial, and as far as we know has not been developed further in the interim. LFG-DOP adds robustness to LFG, as well as being used for disambiguation purposes. Furthermore, the probabilities derived are real numbers based on the treebank from which they are derived—in OT-LFG, the numbers are invented. In addition, the OT-constraints have to be manually ordered, whereas alternative LFG-DOP structures come with probabilities associated with them in the first place. Finally, OT’s use as a preference mechanism in ranking LFG analyses is quite different from the original intention of OT, namely arriving at an optimal order of a set of constraints so as to demonstrate the grammaticality
of certain strings, while ruling out others.

4.3 Restriction and Discard

We described in section 3.2.2 how Kaplan & Wedekind (1993) introduced the Restriction operator in an attempt to deal with some difficult translation cases such as embedded headswitching. It is of interest to note that the Discard operation of (Bod & Kaplan, 1998) is implementable in terms of Restriction. Consider the LFG-DOP fragment (278):

(278) \[ \begin{align*}
S & \rightarrow \text{NP} \rightarrow \text{VP} \\
\text{Kim} & \rightarrow \text{SUBJ} \rightarrow \text{PRED} \rightarrow \text{NUM} \rightarrow \text{SG} \\
\text{TENSE} & \rightarrow \text{PRES} 
\end{align*} \]

Compare fragment (278) with the LFG-DOP fragment (279):

(279) \[ \begin{align*}
S & \rightarrow \text{NP} \rightarrow \text{VP} \\
\text{Kim} & \rightarrow \text{SUBJ} \rightarrow \text{PRED} \rightarrow \text{NUM} \rightarrow \text{SG} 
\end{align*} \]

We see that (280) has been discarded from (278) to form (279):

(280) \[ \begin{align*}
\text{TENSE} & \rightarrow \text{PRES} 
\end{align*} \]

In terms of restriction, if \( f = (278), \) and \( g = \text{PRES}, \) then \( f \backslash \text{TENSE} g \) is equal to f-structure (281):

(281) \[ \begin{align*}
\text{SUBJ} & \rightarrow \text{PRED} \rightarrow \text{NUM} \rightarrow \text{SG} \\
\text{‘Kim’} & \rightarrow \text{SG} 
\end{align*} \]

Both Restriction and Discard have the effect of throwing away attributes and values. The difference is that Discard operates freely, whereas with Restriction one has to specify exactly which attribute is to be thrown away, and thus one can correlate the throwing away of a particular attribute with some other effect or property. Kaplan & Wedekind (1993) do this to describe how the semantics of the unrestricted f-structure is related to the semantics of the restricted one (cf. (119), p.83).

Given this, it is possible to see that Discard could be implemented in terms of Restriction, by constructing a set of f-structure attribute pairs (e.g., \( f \text{TENSE}, \) \( f \text{SUBJ}, \) ...) and stating that Discard maps from the powerset of that set, throwing away all features that a particular element of the powerset includes. While of interest from a theoretical point of view, in practice this may not turn out to be a helpful way of defining Discard, since additional machinery needs to be introduced in order to specify the arbitrary behaviour of Discard. We leave further speculation on this to other interested parties. If nothing else, it suggests perhaps that Discard and Restriction should be integratable without too many difficulties, should it transpire that
both operations are required. Given that our final model of translation, LFG-DOT4, relies significantly on
the Discard operation both for robustness as well as to avoid the problems of limited compositionality in
DOT2 and LFG-DOT3, we would hope in future work to investigate whether a combination of Discard and
Restriction may facilitate a more optimal solution.

4.4 The Discard Function in LFG-DOP

We consider here two possible interpretations of how Discard affects the treebank: the first possibility (P1)
assumes the machinery of LFG-DOP in its entirety as outlined by Bod & Kaplan (1998); the second, novel
possibility (P2), whilst maintaining the Discard operation, employs it solely at what might be termed ‘parses-
time’.

In P1, the treebank contains all possible fragments derivable via Root, Frontier, and Discard. That is, the
fragments derivable via Discard are created prior to combination, which follows on in a separate procedure.
The immediate criticism which arises in such circumstances is that the treebank is potentially huge, and
perhaps unmanageable. Discard is used solely to improve robustness, in case we encounter ill-formed strings,
or well-formed strings whose interpretations can only be obtained via the generalization of f-structure frag-
ments in the treebank. Note also that the adoption of this model will also affect combinations where there
is no ill-formed input, in terms of increasing the number of potential candidate fragments participating in
the combination stage, which causes the probability model to be altered. A potentially far more serious
problem is the increased processing time necessary to cope with the larger bag of fragments. The good thing
about maintaining such a huge treebank is that no extra routines are needed (although the search problem
is potentially an astronomical one).

However, already for Tree-DOP (Bod 1995) the number of fragments becomes extremely large even for
relatively small corpora, e.g. 10^8 fragments for the (small, at 750 sentences) ATIS\textsuperscript{11} corpus if lexicalized
fragments are used.\textsuperscript{12} Despite this, it was possible to make Tree-DOP practical, first by estimating the most
likely parse via Monte-Carlo sampling (\textit{op cit}, then by estimating the most likely parse via the most likely
derivation (Sim'an 1995, 1996). Although in this latter case the results are sub-optimal compared with
those obtained ‘full’ processing, the parser created operates in near to real time for a corpus of 10,000 strings
of \textit{OVIS} data.

Nevertheless, if there is an explosion of fragments in DOP, then the number of fragments in LFG-DOP is
potentially crippling. We now turn our attention to investigating ways of limiting the scope of Discard to
cut down on the number of fragments produced.

4.4.1 Limiting the Number of Fragments

Despite the potential explosion of fragments in LFG-DOP, a number of ways of limiting this suggest them-

\textsuperscript{11}http://www.cis.upenn.edu/ldc/ldc\_catalog.html\#atis
\textsuperscript{12}We show in section 5.2.1 (cf. Figure 5.2, p.165 and Figure 5.3, p.166) that in some cases DOT and LFG-DOT translation
corpora are smaller than their respective DOP and LFG-DOP monolingual equivalents.
1. Cormons (1999:85) suggests that it is possible to distinguish between lexical features such as gender and number on the one hand, and structural features like case on the other\textsuperscript{13}. Lexical features can only be discarded if the word (the PRED value) to which they are linked is also discarded.

2. Those tree fragments greater than depth 1 (i.e. containing some categories with no associated terminals) could be disregarded unless at least one non-terminal contains an overt lexical item as its daughter. That is, they should be in a generalized version of Greibach (rather than Chomsky) Normal Form.\textsuperscript{14} Greibach NF always has a terminal symbol as the leftmost daughter, but we do not require such a strict imposition, rather that there be a lexical item somewhere in the tree. We discuss this lexicalization issue further in section 6.3.6, especially with respect to its implications for translation models, and LFG-DOT in particular.

The problem in these circumstances is that we would most likely need to incorporate an adjunction operation into our DOP model (turning it into probabilistic approaches to Tree-Adjoining Grammar, e.g. PTAG—Resnik 1992; SLTAG—Schabes 1992), as it is known that even if lexicalized CFGs can generate the same strings, they severely undergenerate with respect to structures. This would have unfortunate consequences for our models.

3. We might attempt to redefine the Discard function so that its effects are not so wide-ranging.

It is this latter proposal that we now explore further.

4.5 A Reinterpretation of the Discard Function

We may hope that the savings in processing time for DOP (cf. section 4.6) carry over to LFG-DOP MT. Assuming this not to be the case, however, we advocate the adoption of P2, which strives to avoid such extra processing. Discard is used to derive fragments only where absolutely necessary, i.e. in those cases where generalized fragments are the only recourse to achieving an analysis given the treebank derived via Root and Frontier. The treebank is, therefore, much smaller than in P1. P1 is simple, but perhaps intractable due to the search problem. In any case, we showed in section 4.1.1 (cf. (234)-(238)) that Discard-generated fragments occupy too much of the probability space, leading to smaller probabilities for certain analyses via well-formed fragments (cf. also section 6.3.6).

If we adopt P2 to try to avoid this, then the minimum criterion is that Discard should apply only when unification fails. This entails further computation, as we need to stipulate what ‘unification failure’ means. For example, we want Discard to operate when we have a clash in value for a given attribute such as NUM=SG and NUM=PL, but not when we have a clash of PRED values, such as PRED=John and PRED=Mary. There must be a countable number of instances of such cases—one can think of subject-verb agreement,
relative-clause agreement between the verb and the modified NP, as well as between this NP and the relative pronoun. Others will include all cases where there is a chain of derivation, i.e. including all movement phenomena, and so on. Once this list has been established, we envisage three ways in which composition via Discard could work:

1. Use Discard every time any such unification failure is encountered.

   The problem here is that there would be lots of redundancy, and the amount of computation is potentially crippling. The advantage is that such computation is done on the fly, so that no storage of the extra fragments derived would be needed.

2. For all structures in the treebank affected by such a unification failure, run Discard over all such structures, and put the results into an ‘ill-formed bag’ (IFB), leaving the original treebank unaffected. That is, nothing goes into the ‘well-formed bag’ (WFB)—the original treebank—which has been produced by a Discard operation.

   The problem with this scenario is that there is a lot of computation, but it only needs to be done once for each type of unification failure. The merit of such an approach is that as a distinction is made between what is and what is not grammatical, the separate IFB is available for subsequent ill-formed input to use, making it more probable that a ‘correct match’ (or at least some useful fragments of structure) will be found for these, analogously to EBMT (cf. sections 1.2.3 and 5.1). Note also that the definition of grammaticality with respect to the corpus (cf. chapter 4) is maintained here.

3. For all structures encountered subsequent to a unification failure, that would be affected in the same way, run Discard over all such structures in the treebank, and calculate probabilities on these affected structures only.

   The main problem with this approach is that all structures already encountered which have had unification failures need to be stored, so that other similar instances can be identified in the future. The main advantage is that the whole treebank is not affected by the Discard operation; only ‘similarly affected’ structures are subjected to the generalization process.

No matter which option is selected here, we have assumed (unrealistically) that the treebank size remains constant, i.e. there is no dynamic addition to the treebank. If this were to happen, i.e. if we add a new tree and associated f-structure (and the fragments derived via Root and Frontier) to the treebank, in theory we should be able to delete some fragments from the IFB. However, in practice this is not a problem because if a hierarchical model is posited in which the WFB is searched before the IFB, the new well-formed fragments in the WFB will obviate the need to consult the IFB at all in such cases, although there may be an element of redundancy in such a model. We discuss such hierarchical models at the end of section 4.6.

One more urgent problem is the definition of the circumstances whereby the IFB is to be visited. We propose that once the WFB has been left, it is impossible to return until the analysis of the next sentence. This restriction is necessary because sentences can contain more than one unification error, so that if one tries to resolve the first error encountered by going to the IFB, when trying to resolve subsequent errors, one would need to return to the WFB, thereby potentially bringing back fragments derived from the IFB into
the WFB, and we want to keep this pristine. The stipulation is, therefore, that one only goes to the IFB when a fragment with Root=ns has been fully evaluated in the WFB.

It is clear that the Discard function of Bod & Kaplan (1998) is far too unconstrained, so that its application needs to be limited in some way to control the foreseen explosion of fragments. In much the same way as is suggested here (cf. Way, 1999), Bod proposes a solution whereby the fragments produced via Root and Frontier are separated from the Discard fragments. He then uses Good-Turing to ensure that the well-formed fragments produce derivations that are favoured over the ‘ungrammatical’ Discard fragments. We discuss this further in section 6.3.6.

4.5.1 The Role of Discard in each Stage of the Translation Process

Finally here, let us convince ourselves of the contribution of the Discard function to the robustness issue. In traditional MT systems, as noted in chapter 1, three main distinctions are made between different phases of the translation process—parsing, transfer, and generation. Although we do not describe our LFG-DOT models of translation until chapter 6, it is useful to ponder how Discard improves robustness in each of these phases.

The first of these—parsing—needs little further discussion, as this is the primary reason for its importation in the first place: namely, parsing ill-formed input, and dealing with unknown words and structures. Of course, to the system these are one and the same. So, with respect to MT, in dealing with the source language the task of the syntactic disambiguation component is the calculation of the probability distribution of the various probable parses. Bod (1995; 1998) outlines possible methods of dealing with unknown words (cf. sections 2.2.1 and 2.2.2), where Tree-DOP removes terminals from trees to at least allow POS categories to be estimated. This helps provide a (hopefully correct) context for DOP to allow further processing to continue. However, Bod (1995) notes that the biggest problem is not the parsing of unknown words, but rather the processing of items which are contained in the corpus, but for which other grammatical categories are required (unknown category words, cf. section 2.2.2).

In the transfer phase (that is, where the two languages interface with one another in the system), the effect of Discard in parsing has knock-on effects here too. If an interpretation for some unknown element has been correctly assigned (i.e. surpasses some user-defined threshold, say), then this should help ensure the correct selection of the lexical item on the target side. Given also that all source fragments will be aligned with their target counterparts, some translation will always be found, if all source words are contained in the corpus. This may not, however, be the correct translation owing to the possibility of other translations for ambiguous words not (yet) covered in the corpus, or even lexical gaps. In all such situations, of course, rule-based methods have nothing to say.

When it comes to generating the target string, if we incorporate a target language LFG-DOP model into our translation models then Discard contributes both on the source and target sides in a similar manner.
4.5.2 A Summary of the Two Interpretations of the Discard Function

The adoption of P1 (i.e. using Bod & Kaplan’s (1998) formulation of Discard) has a major problem when it comes to searching the treebank because of the explosion of fragments, which we note is a particular problem for LFG-DOP (cf. section 4.1.1, where we make some observations on the impact of Discard on corpus size, and section 6.3.6, where we discuss the impact of Discard-generated fragments on our probability models). We therefore propose to adopt (some variant of) P2. However, there are many questions to be resolved before doing so. Nevertheless, if we adopt P2, the benefits are:

1. The problem of explosion of fragments we get via P1 is alleviated considerably, i.e. the treebank (WFB) stays the same size. In addition, P1 adversely affects the processing of well-formed input, which P2 avoids.

2. Depending on the number of types of ill-formed input (i.e. the small number which cause unification failure), the IFB might not be too big, so the added complexity of the hierarchical model introduced here may not be too onerous compared to the vast reduction in the number of fragments assumed in P1.

3. Just as with EBMT systems, the larger the IFB, the greater the chance of finding a good match for the ill-formed input one might be confronted with in the future, which may cut down the amount of processing required by the Discard operation as the size of the treebank increases.

4.6 Improving the Efficiency of DOP-based models

Despite DOP models outperforming other statistical methods, the major criticism of DOP is that it is notoriously inefficient: “One of the most inefficient performance models of language is the DOP model” (Sima’an, 1999:9). We have given some consideration to this problem in the previous sections. It is appropriate to hypothesize as to how DOP-based models in general, and those based on LFG-DOP in particular, can be made more efficient.

Why is DOP so accurate, yet at the same time so inefficient? These two characteristics are intrinsically bound together: SCFGs can only produce a parse in one way, but STSGs may generate the same parse using various different derivations. Scha (1990) noted that this ‘redundancy’ was an essential component of the DOP model. Indeed, the probability of a parse is calculated in DOP by summing the probabilities of each of its (unique) derivations. Sima’an points out that:

“DOP models do not account for two appealing properties of human language processing: firstly, that more frequent utterances are processed more efficiently, and secondly, that utterances in specific contexts, typical for limited domains of language use, are usually less ambiguous than they are in general contexts” (Sima’an, 1999:1, original emphasis).

Indeed, in one of the earliest expositions on DOP, Scha notes:
“We expect that, in the present processing model, the most plausible sentences can be analyzed with little effort, and that the analysis of rare and less grammatical sentences takes significantly more processing time” (Scha, 1992:16; translation by Sima’an (1999:61)).

Grammaticality in DOP is associated with frequency in the treebank, and the notion of ‘plausible sentence’ is only interpretable through its probability models, based again on frequency in the treebank. The first of these points—that more frequent utterances ought to be processed more efficiently—is achieved in DOP via Monte-Carlo parsing. The second is entirely appropriate for a model of translation, as it is well known that MT works best when geared towards a restricted area of sublanguage (cf. section 1.2.3). Furthermore, Sima’an points out (op cit., p.12) that “more frequent input in limited domains can be processed more efficiently if language use in limited domains is modeled as unambiguously as possible”. We observe that both DOP and LFG-DOP being predicated on treebanks15—by their very nature unambiguous—ensures this lack of ambiguity and therefore enables more efficient treatment of the sort suggested by Sima’an.

These properties form the basis for significant improvements in the efficiency of DOP models. Sima’an states (op cit., p.184) that “it should be possible to make processing time depend on general properties of the distribution of sentences in some domain of language use, rather than only on properties of individual sentences” (original emphasis). There are two main sources of inefficiency for DOP:

- the huge STSGs produced, given any reasonably sized treebank;
- the complexity of the disambiguation process using STSGs.

As has already been pointed out, Goodman (1998) shows that Monte-Carlo parsing is exponential-time. Sima’an provides a proof that probabilistic disambiguation under SCFG/STSG is NP-complete.16 Given that, he states that:

“The proof ... implies that for efficient solutions to these problems it is necessary to resort to non-standard and non-conventional methods, as:

- to model observable efficiency properties of the human linguistic system ... by employing learning methods prior to probabilistic parsing,
- to approximate the DOP model by allowing more suitable assumptions,
- to approximate the search space delimitied by an instance of any of these problems through ‘smart’ sampling to improve on the brute-force Monte-Carlo algorithm (Bod, 1993a; Bod, 1995)” (Sima’an, 1999:58).

Sima’an’s solutions are twofold:

1. an off-line method, which ‘specializes’ the domain;

15A nice observation by Sima’an (1999:7) is that using treebanks as training data can be viewed as supervised learning, whereas using unparsed corpora as training data constitutes unsupervised learning.

16For SCFG, disambiguation is NP-complete for word-graphs, not for sentences (Bod, personal communication).
2. an on-line performance model, which acquires less ambiguous and more efficient probabilistic grammars from the ‘specialized’ version of the treebank.

The intention behind (i) is to reduce the inherent ambiguity in the treebank by grammar specialization,17 by which a new, less ambiguous, grammatical description is obtained, “which, in turn, may serve as the basis for new, smaller, probabilistic models” (op cit., p.61). The algorithm presupposes that it is possible to span a good approximation of the parse-forest (i.e. treebank) of an STSG using a relatively small CFG, which is a simplification of the CFG underlying the DOP STSG.18 He also points out that probabilistic disambiguation is by far the main source for time and space consumption, observing that “typical grammars span the parse-space of a sentence with little cost (time and space)—some 1% of the total cost of parsing and disambiguation—relative to the cost of probabilistic disambiguation” (ibid). The second phase performs “on-line full-disambiguation. The DOP model is the most suitable candidate for this task since it generalizes over most existing probabilistic models of disambiguation” (op cit., p.38).

Grammar specialization can be compared and contrasted with techniques which dynamically prune the search space (Rayner & Carter, 1996; Goodman, 1998). The major problem with pruning is that one needs to generate the complete hypothesis space before pruning can be undertaken, and it is clear that generating analyses and then pruning them is a time-consuming endeavour which grows with the input. Sima’an states (Sima’an, 1999:61) that “pruning does not provide a solution for the problem of grammar redundancy. Nevertheless, in practice pruning techniques can complement off-line specialization methods as (Rayner & Carter, 1996) show”. Samuelsson (1994) uses a completely data-driven function to define the criteria under which specialization is to operate. He computes the entropy of each node, and using thresholds, ‘cuts off’ (prunes) nodes which have a low entropy, i.e. which have a low information content, indicating locations in the tree which are expanded using a predictable set of rules. The basic idea is to investigate to what extent corpus compilation methods can increase parsing efficiency, without reducing coverage. In the same spirit, a more recent paper performs experiments on specialization of LFG grammars (Cancedda & Samuelsson, 2000), where on large LFGs speedups of a factor of 6 were observed. By introducing a second stage into the process, no loss of coverage resulted: if the specialized parser failed to process a particular string, this was then sent to a parser equipped with the original grammar.

It is clear that grammar ‘specialization’ conserves the tree-language coverage (cf. also section 4.7). Perhaps surprisingly, Sima’an expects that “the specialization of DOP models should not result in unexpectedly worse accuracy results than the original DOP models; in fact, when the amount of training material is sufficient we expect that the ARS should not result in any worsening of precision at all” (op cit., p.140, original emphasis). Specialization gives a significant improvement in time and space consumption over the original treebank grammar.

Most importantly, Sima’an’s algorithms have time-complexity linear in STSG size and cubic in input length. Furthermore, this is achieved without sacrificing memory-use, which Sima’an notes is “a typical situation in all other alternative algorithms that could be used for parsing DOP” (op cit., p.107), such as the CKY parser (Younger, 1967), the Earley parser (Earley, 1970; Schabes & Joshi, 1988), and the Stochastic Lexicalized

---

17Sima’an calls his theoretical framework Ambiguity Reduction Specialization (ARS).

18One might presume that such a technique might exacerbate the effects of sparse data, but as far as we can tell, Sima’an does not address this issue in his thesis.
CFG parser (Schabes & Waters, 1993). With respect to the CKY algorithm, Sima’an argues that there are two main disadvantages, namely a dramatic expansion in grammar size and the production of a large number of ‘useless’ items which cannot contribute to the overall sentence parse. For real-life DOP STSGs, these disadvantages “imply either very slow speeds or very huge memory costs or both” (Sima’an, 1999:112). Sima’an also notes (op cit., p.106) that these algorithms are most suitable for large DOP STSGs: their superiority is enhanced as the ratio between the size of the DOP STSG and the size of the CFG underlying it becomes larger ... [and that they] compute the exact values rather than approximations as Monte-Carlo does”. Furthermore, “the speedup that these algorithms achieve (in comparison to Monte-Carlo) is in the order of magnitude of hundreds of times (due to the optimized deterministic polynomial nature of the algorithms rather than any implementation detail)” (op cit., p.133).19 Importantly, he asserts that “the present algorithms are ‘pure’ solutions, in the sense that they do not incorporate any assumptions or approximations that do not originate from the DOP model” (op cit., p.104).

Despite such optimizations, a number of heuristics are brought into Sima’an’s experiments to enable still greater efficiency. We described earlier that Bod’s (1995) experiments pruned the search space by limiting subtree-depth. Bonnema & Scha (1999) point out that limiting this alone is not effective enough. Sima’an observes that “an interesting aspect of the MPD (most probable derivation) in DOP is a general tendency for preferring shorter derivations involving more probable trees. Shorter derivations imply a smaller number of substitutions ... This is operationalized through setting an upper bound on the number of substitution-sites a DOP STSG’s elementary-tree is allowed to have: for example, a maximum of 2 substitution-sites per elementary tree. This heuristic ... turned out to reduce the size of the DOP STSG up to two orders of magnitude without loss of accuracy or coverage” (Sima’an, 1999:132, original emphasis). Other parameters which are varied (only one per experiment) include:

- placing an upper-bound on the number of terminals per elementary-tree;
- placing an upper-bound on the number of consecutive terminals per elementary-tree;
- limiting the depth of fragments;
- limiting the number of substitution sites.

One constant is that subtrees of depth 1 are allowed to be elementary-trees of the projected STSG notwithstanding the other constraints. Sima’an provides results on a number of levels for experiments on the OVIS and ATIS treebanks:

- Percentage of sentences recognized;
- Depths at which the results stabilize;
- Tree-language coverage;
- Speed.

19 Although in parsing the OVIS treebank, the speedup factor is ‘only’ 10 times compared to DOP.
The *OVIS* treebank contains 10,000 syntactic and semantic trees, which are user utterances in response to questions asked by the dialogue system. The average sentence length is 3.43 words, but Sima’an performed experiments on sentences with a minimum length of 2 words, making the average length in his experiments 4.57 words. For comparison’s sake, Sima’an performed these experiments on DOP (upper bound depth 4, henceforth *DOP*¹), specialized DOP (SDOP, upper bound depth 2, henceforth *SDOP*²), and ISDOP, an integration of SDOP with the specialized grammar parser (a partial parser, hence PARDOP) and the CFG underlying the original DOP STSG. ISDOP is a two-level model, and works as follows: parse the input with PARDOP. If the parse-space is complete, then disambiguate using SDOP. If the parse-space is incomplete, then apply the underlying CFG to complete the parse-space, and disambiguate using the DOP STSG.

*DOP*⁴ and *SDOP*² were chosen as depth 4 gave the best results for the DOP model, and *SDOP*² gave the smallest models which had comparable results, though not the best results achievable with SDOP models.

The results can be summarized as follows:

- for a treebank of 9,000 trees, the size of the SDOP STSG was only 17% of the corresponding DOP STSG, resulting in a 5.7 times reduction in memory consumption;
- the ISDOP models recognize (almost) as many sentences as DOP, but SDOP recognizes fewer strings. But as the size of the treebank increases, the gap narrows from 9% to 3%;
- SDOP and ISDOP are faster than DOP, with a speed-up of 2.25 times for a treebank of 2,000 trees to 4 times for a treebank of 9,000 trees;
- in experiments varying the depth of subtree allowed, SDOP models stabilize at depth 6, whereas DOP models are still growing at depth 10. Moreover, the largest SDOP model is less than half the size of the largest DOP model;
- much smaller and faster SDOP and ISDOP models achieve a comparable accuracy comparable to deeper DOP models, with ISDOP⁺¹ (i.e. integrating a DOP STSG max. depth 1—the superscript—with an SDOP STSG max. depth 1—the subscript) having an exact syntactic match comparable to DOP³ while being 4.8 times faster.

Sima’an concludes from these results that “ISDOP and SDOP models achieve as good accuracy results as DOP models but ISDOP and SDOP models achieve these results much faster and at much smaller grammar-size” (Sima’an, 1999:158). Interestingly, however, when carrying out experiments on *OVIS* with the semantic analyses stripped out, Sima’an found that “the specialization effort here is less successful than it is on the *OVIS* tree-bank with full annotation” (*op cit.*, p.161). Although we do not explore this further in this thesis, we note that Sima’an’s finding may require more complex LFG models including semantic levels of description (e.g. Kaplan & Wedekind, 1993; Butt 1994; cf. section 3.2.1 above) to be considered as LFG-DOP (and, consequently, LFG-DOT) models of language processing.

Furthermore, Scha’s desideratum that more frequent input be processed faster is satisfied in an experiment where *ISDOP*⁺² and *DOP*⁴ were adapted to parse and evaluate POS-tags rather than strings. While the recognition power of ISDOP and DOP was comparable, the specialized model was much faster than for

---

²⁰More recent results are available on 40,000 sentences of the WSJ, cf. Bod (2006a); Sima’an (2000).
DOP, consumed less memory-space, and generally speaking processed more probable input faster than less probable input. DOP, on the other hand, behaved rather more unpredictably.

Experiments on the ATIS treebank, however, were less successful. The average sentence length here is 8.2 words, and the number of trees 13335, rather longer than for OVIS, thereby rendering the tasks in hand somewhat more difficult. For instance, Sima’an notes that “it takes more than 10 days for DOP to parse and disambiguate the test set (1000 sentences)” (op. cit., p.168).

Summarizing the results again, Sima’an observes that:

- compared with a DOP model containing all subtrees in the treebank, SDOP is substantially smaller, but restricting the depth of subtree permitted renders SDOP smaller only from depth 3;
- recognition power and accuracy is comparable between SDOP and DOP;
- with respect to accuracy and speed, the results for SDOP fit somewhere in between those for DOP and DOP

Sima’an observes that “the ATIS experiments have exposed a weakness in the current implementation of the specialization algorithm: it does not reduce ambiguity substantially to allow speed-up” (op. cit., p.172). He also notes that “all models tend to assign too much of the probability mass to extremely large elementary trees” (op. cit., p.156), which explains the degradation of his SDOP and ISDOP models: SDOP models do not include as many small elementary trees as DOP models which obviously means that they assign a larger probability mass to very large trees. This observation leads Bonnema et al. (1999) to investigate new probability models for DOP which take the frequency with which fragments participate in other trees as well as their relative frequency into account (cf. section 6.3.5).

Finally, Sima’an notes that “we cannot conclude that these optimizations have completely solved the efficiency problem for DOP ... Although our optimized algorithms enable a substantial speed-up when compared to those described in (Bod, 1993a), alas both kinds of algorithms are currently not yet useful for practical applications in the ATIS domain” (op. cit., p.184).

### 4.6.1 LFG-DOP and Efficiency

It is reasonable to expect that some of Sima’an’s techniques may carry over to improving the efficiency of LFG-DOP models, including LFG-DOT. For instance, if we wish to avail ourselves of some of the speed-up capability introduced by Sima’an, then the best way forward may be to have a multi-level LFG-DOP (and LFG-DOT—in chapter 6 we postulate an ‘extended transfer’ level for our fourth, and final, model of translation) model:

1. Specialized LFG-DOP model;
2. The original LFG-DOP model, with all original trees;
3. IFB (with Discard).
That is, we may first employ grammar reduction on the treebank to cut down on the number of \((c,f)\) fragments and improve the probability model accordingly, particularly as it is envisaged that we translate in a restricted domain. We may then use the full set of original \((c,f)\) fragments “only when it is sure that the input sentence is not in the language of the SDOP’’ (Sima’an, 1999:91, original emphasis). Finally, we have recourse to the set of generalized fragments in the IFB via Discard. Such a multistratal model may be used in parsing the source or target languages, and may also facilitate transfer via the \(\tau\)-equations if we keep grammatical fragments distinct from (potentially) ill-formed ones (cf. Bod (2000a, section 6.3.6). Indeed, if ambiguity reduction is to be used, Sima’an (\textit{op cit.}, p.69) observes that “we assume that the tree-bank contains only the correct structures”. That is, SDOP operates on the set of ‘correct’ DOP (or LFG-DOP, accordingly) fragments.

In combination with grammar specialization and restricting the scope of Discard, it may be that insights from parsing lexicalized grammars (cf. Eisner and Satta, 1999; 2000) may help to parse grammars like LFG more efficiently. For example, subtrees are indexed in the XTAG system in order to rule out certain analyses even before parsing begins. Each subtree node which dominates a lexical item is indexed for what sorts of structures (and words) may appear to its left and to its right. In terms of LFG f-structures, what may happen in such cases is that rather than bringing about a reduction in the number of fragments, we may actually increase the number of analyses as unification implies a growth in the number of analyses. What would probably be required, therefore, is that unification of f-structures be restricted to ‘substitution’ (i.e matching of features).

Furthermore, we might also give some consideration to pruning the number of fragments in an LFG-DOP treebank. LFG-DOP models are more informed than DOP models, so they ought to lead to more informed pruning (cf. Charniak, 1997; Collins, 1997; Goodman, 1997). This work has its roots in ‘bilexical’ grammars, in which each grammatical rule is specialized for two individual words (to capture collocations, for example). Collins’ (1997) work shows how subcategorization frames may be incorporated into a statistical parser. Sima’an (personal communication) thinks that using such techniques may bring about good pruning, in which case LFG-DOP ought to work at least as fast as DOP.

We do not investigate such techniques further in this thesis, but raise such issues here as possible future work. What can be said with some confidence is that DOP-based models can no longer be tarred with the inefficiency brush as has happened in the past, and we expect the concerns of efficiency to be met more fully in the future.

### 4.7 Semi-Automatic Generation of LFG-DOP Corpora

We have noted that the large, high quality training corpora with both tree and feature structure annotations required for processing language with LFG-DOP do not currently exist. Manual construction of such corpora is time consuming, error prone and expensive. A new method for automatically annotating treebanks with feature structure information has been developed (Van Genabith \textit{et al.} (1999a,b,c); Sadler \textit{et al.} (2000); Frank \textit{et al.} (2001)). The basic idea is simple and effective: firstly, the CFG is automatically extracted from the treebank following the method of (Charniak, 1996); secondly, a set of underspecified feature annotation...
principles is developed using regular expressions; thirdly, the principles are compiled over the extracted CFG; and fourthly, the annotated CFG is used to ‘reparse’ original tree annotations in the treebank to induce a feature structure. The reparsing process is deterministic on the original tree representations.

The method outlined above was tested on the publicly available set of 100 sentences of the parsed AP treebank.21 Despite its small size, this was sufficiently large to demonstrate the plausibility of their approach. An example sentence from the treebank is (282):

(282) A001 39 v
      [N The_AT march_NN1 N][V was_VBDZ [J peaceful_JJ J]V].

For the sentence in question, the march was peaceful, the CFG-rules and lexical items which are automatically extracted are those in (283):

(283) lex(at(the)).       rule(n(A), [at(B),nn1(C)]).
    lex(nn1(march)).     rule(j(A), [jj(B)]).
    lex(vbdz(was)).      rule(v(A), [vbdz(B), j(C)]).
    lex(jj(peaceful)).   rule(s(A), [n(B), v(O)]).

Treebank tagsets often use a very fine-grained set of category labels, and these can be used to provide f-structure information pertaining to that class of words. Van Genabith et al. (1999a,b,c) associate a macro with each lexical category type in the AP tagset. The appropriate macros for the example sentence in (282) are shown in (284):

(284) macro(at(Word),FStr) :-
    FStr:spec == Word.
macro(jj(Word),FStr) :-
    FStr:pred == Word.
macro(nn1(Word),FStr) :-
    FStr:pred == Word,
    FStr:num == sg.
macro(vbdz(_Word),FStr) :-
    FStr:tense == past,
    FStr:pred == be.

The extracted rules in (283) are then annotated with LFG functional schemata by hand. The annotated rules are those in (285):

21http://www.hit.uib.no/icame.html
(285) \[
\text{rule}(n(A), [at(B), nn1(C)]) :- \quad \text{rule}(v(A), [vbdz(B), j(c)]) :- \\
A \equiv B, \\
A \equiv C.
\]

A ‘reparsing’ interpreter then recursively reparses the annotated treebank entries (not the strings) by following the original tree annotations provided. In so doing the interpreter solves the constraint equations associated with the grammar rules and lexical macros involved in the parse, returning a single f-structure. For the march was peaceful, the output of the reparsing interpreter is given in (286):

(286) \[
sent(n(at(the), nn1(march)), v(vbdz(was), j(jj(peaceful))))
\]

\[
\begin{align*}
\text{subj} : & \text{spec} : \text{the} \\
\text{pred} : & \text{march} \\
\text{num} : & \text{sg} \\
\text{xcomp} : & \text{pred} : \text{peaceful} \\
\text{subj} : & \text{spec} : \text{the} \\
\text{pred} : & \text{march} \\
\text{num} : & \text{sg} \\
\text{tense} : & \text{past} \\
\text{pred} : & \text{be}
\end{align*}
\]

The f-structure (286) illustrates how re-entrancies can be enabled quite successfully by annotating rules such as (285), rather than inserting such equations in the lexicon (cf. (256, p.131, and (274), p.138) and resultant discussion). The equation B:subj \equiv C:subj in (285) states that the subject of the verb (vbdz) is is identical to the subject of the xcomp (open complement) headed by the adjective (j) peaceful.

Given the complexity of some of the strings and accompanying structures, producing f-structures automatically in this manner may be easier than producing them by hand on the fly, for a corpus of any real size. For example, for the 100 sentences used in the experiments of Van Genabith et al. (1999a,b,c), the treebank fragment contained more than 2,000 words and more than 1,300 phrasal subtrees, from which more than 500 grammar rules and 700 lexical entries were induced. The longest sentence in the fragment contains 43 words, the shortest 4, with an average of about 20, but all sentences are real examples of AP newswire, containing a wide array of linguistic constructs, including traditionally difficult constructions such as conjunctions. Anyone working in corpus-based techniques remains open to the vagaries of the coding of that corpus, so that misparses cause incorrect f-structures to be produced, but few other errors are introduced by the automatic procedure of Van Genabith et al. (1999a,b,c).
<table>
<thead>
<tr>
<th></th>
<th>1 analysis</th>
<th>2 analyses</th>
<th>4 analyses</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-Deterministic Grammar</td>
<td>67</td>
<td>27</td>
<td>6</td>
<td>100</td>
</tr>
<tr>
<td>Non-Deterministic Grammar with semantic-form filtering</td>
<td>93</td>
<td>7</td>
<td>0</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 4.4: Results of ‘reparse’ the AP Treebank to produce F-structures

In a subsequent stage, and as a first step towards providing a stand-alone LFG resource, Van Genabith et al. (1999a) used the treebank entries themselves to automatically provide LFG semantic forms. Thus semantic forms are induced from the f-structures, just as the CF-PSG is induced from the original treebank. As an example, consider the treebank entry in (287) and its associated f-structure:

(287) \[ \text{tree(a001,'43'),v,sent(n(nn2(police)),v(vvd(check),n(at(the)),nn1(coliseum)),p(if(for),n(nn2(bombs)),p(ics(before),n(at(the)),nn1(march))))}. \]

From the f-structure for the sentence in (287), the semantic form compiler produces the semantic forms in (288):

(288) police(\[\]) coliseum(\[\]) bombs(\[\]) march(\[\]) for(\[\]) before(\[\]) checked(\[\])

Given the structural analyses for the 100 sentences originally provided by the hand annotators of the treebank, and their own manual annotation of the resultant (highly specific) rules to produce f-descriptions, the extraction of semantic forms from f-structures created on the basis of these resources produced reasonably correct (if somewhat incomplete) results. These semantic forms can then provide a treatment of subcategorization for subsequent free text parsing.

One area which is particularly fraught with difficulties is that of distinguishing between obliques (which are subcategorized for) and adjuncts (which are not). In the original grammar in Van Genabith et al. (1999c), for each rule in question a decision was made as to whether a prepositional phrase should be analysed as an adjunct or as a subcategorizable grammatical function based on analysis of occurrences of the rule in question in the original corpus. This approach yields surprisingly good results (because the rules extracted from the treebank are highly specific), but is not a general solution.

Consequently, a second experiment was performed in Van Genabith et al. (1999b) in which a limited use of disjunction in the rule annotations for four rules (less than 1% of the grammar) is introduced, and the best analysis in the case of duplicate f-structures is selected manually. Semantic forms are then derived from the set of ‘good’ f-structures. This clearly represents a further level of manual intervention, but produces better results. The results for the two grammars (with the well-formedness constraints completeness and coherence enforced) are shown in Table 4.4.

In order to produce target f-structures, what is ultimately required is that \(\tau\)-equations be added to semantic forms such as (288) and structural rules such as (285), and the original treebank entries reparsed. For example, the German target f-structure corresponding to the input sentence in (282) is (289):
(289)  \[
\begin{align*}
tau &: \text{subj} : \text{spec} : \text{die} \\
\text{pred} &: \text{demonstration} \\
\text{num} &: \text{sg} \\
\text{xcomp} &: \text{pred} : \text{ruhig} \\
\text{subj} &: \text{spec} : \text{die} \\
\text{pred} &: \text{demonstration} \\
\text{num} &: \text{sg} \\
\text{tense} &: \text{past} \\
\text{pred} &: \text{sein}
\end{align*}
\]

Once these exist, one of the standard LFG generation algorithms (cf. Wedekind, 1988, 1999; Kohl, 1992; Wedekind & Kaplan, 1996; cf. section 3.2.1 above) can be used to produce target c-structures, and the target strings trivially read off these structures. Alternatively, one of the LFG-DOT models of translation given in chapter 6 could be used to derive translations of new input given such a treebank. We give further consideration to how LFG-DOP corpora might be obtained in section 6.3.8.

4.7.1 Grammar Compaction

Treebank grammars are both large and highly specific. From the Penn treebank, Charniak (1996) derives a grammar of 10,605 rules of which only 3,943 are used more than once in the corpus. An intriguing and significant result of his work is the observation that reducing the ruleset to those 3,943 has very little impact on the quality of the results produced on the test corpus. In general, a significant concern with treebank grammars is that they are rather shallow and use a very large number of tags: the large number of parts of speech permit very many combinations. Some of these will be accidentally missing from the grammar: intuitively, there will be holes or gaps in the grammar derived from a treebank precisely because the tags are so specific. For independent parsing of free text, a grammar is needed which ‘plugs the holes’. That is, a grammar is required which generalizes over the subcategorial instances of combination actually instantiated in the source treebank entries.

As an example, Van Genabith et al. (1999b) show the fine-grainedness of the NP tags in the AP treebank with respect to number, as in (290):
(290) MC cardinal number, neutral for number (two, three..)
MC1 singular cardinal number (one)
MC2 plural cardinal number (tens, hundreds)

NN common noun, neutral for number (sheep, cod, headquarters)
NN1 singular common noun (book, girl)
NN2 plural common noun (books, girls)

NNJ organization noun, neutral for number (co., group)
NNJ1 organization noun, singular (no known examples)
NNJ2 organization noun, plural (groups, councils, unions)

NNL locative noun, neutral for number (is.)
NNL1 singular locative noun (island, street)
NNL2 plural locative noun (islands, streets)

NNO numeral noun, neutral for number (dozen, hundred)
NNO1 numeral noun, singular (no known examples)
NNO2 numeral noun, plural (hundreds, thousands)

NP proper noun, neutral for number (indies, andes)
NP1 singular proper noun (london, jane, frederick)
NP2 plural proper noun (londons, johns, marys)

Most of the subcategorial distinctions drawn in these tags would be expressed by means of grammatical features at f-structure in LFG, as macros will introduce (↑NUM) = , (↑NUM) = SG, and (↑NUM) = PL, as appropriate.

Krotov et al. (1998) note a major problem with a grammar which distinguishes a very large number of tags, namely that there is no indication that the grammar begins to reach completeness. They subsequently compact the grammar they derive from the Penn treebank by eliminating a number of rules which are redundant given that the input they are designed to treat can be handled by other rules. As an example, consider the rule in (291):

(291) NP → DT NN CC DT NN

This may be deleted given the existence of the rules in (292):

(292) NP → NP CC NP       NP → DT NN

While a string which would have been parsed by (291) will still receive an analysis via the rules in (292), it will not be the same analysis. That is, the compaction method of Krotov et al. is not structure-preserving, which may prove to be problematic in certain cases of PP-attachment, for instance, in which different analyses are appropriate.
Consequently Van Genabith et al. (1999b) perform an experiment in compacting the grammar using a set of ‘supertags’ which preserves the original structure of the treebank entries. Compaction takes place after compiling out the lexical macros over the lexical entries, as the fine-grained tags crucially enable partial annotation via the lexical macros. If compaction preceded this stage, a lot of this fine-grained information would be lost—it cannot simply be moved from the lexicon to the rules. Tags are then ‘collapsed’ into a smaller set of supertags. Those for determiners are shown in (293):

(293) AT article (the, no)
   AT1 singular article (a, an, every)
   DB before determiner (capable of pronominal function) (all, half)
   DD determiner (capable of pronominal function) (any, some)
   DD1 singular determiner (this, that, another)
   DD2 plural determiner (these, those)
   DDQ wh-determiner (which, what)
   DDQZ wh-determiner, genitive (whose)
   DDQV wh-ever determiner (whichever, whatever)

<table>
<thead>
<tr>
<th>Grammar</th>
<th>Number of rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original CFG</td>
<td>509</td>
</tr>
<tr>
<td>Max Lex CFG</td>
<td>362</td>
</tr>
<tr>
<td>Max Lex Phr CFG</td>
<td>304</td>
</tr>
</tbody>
</table>

Table 4.5: Compacting the Rules derived from a Treebank using ‘Supertags’

The lexicon and macros are then relabelled along with the treebank, and the relabelled grammar is annotated by manually merging the functional annotations from the original CFG to the collapsed rules. While the length of the RHSs of the rules does not change, this merging process may introduce disjunctions. Reparsing the new treebank entries with the new grammar and lexicon results in a new set of f-structures. In the experiment of Van Genabith et al. (op cit.), two results are given in Table 4.5, as grammars are produced on the basis of a maximal collapsing of lexical tags (the Max Lex CFG) and a maximal collapsing of both lexical and phrasal tags (the Max Lex Phr CFG). However, the additional disjunction introduced by the grammar compaction phase added to the limited non-determinism in the rules and semantic-forms leads to a significant combinatorial explosion of candidate f-structure solutions, as shown in Table 4.6.

This, we feel, shows that the first two of the possible four competition sets (M1 and M2) set out in (Bod & Kaplan 1998, cf. also section 4.1.1 above) are impractical. Having extracted both CF-PSGs as well as
stand-alone LFG grammars, Van Genabith et al. (op cit.) ran two experiments on the same set of 100 AP sentences. The first involved parsing the sentences as free text with the extracted CF-PSG, and stipulating that f-structures were to be unifiable (i.e. Competition set M2 in Bod & Kaplan 1998). As well as taking days to run, certain strings had over 1,000 million \( \langle c, f \rangle \) permissible pairs! The second experiment involved using Competition set M3 in (Bod & Kaplan, 1998), i.e. the same grammars were used (with limited subcategorization information), but the LFG grammaticality checks were employed to prune the number of \( \langle c, f \rangle \) pairs. This time the highest number of \( \langle c, f \rangle \) pairs produced was 96. The full results of enforcing completeness and coherence checks are shown in Table 4.7. This would appear to rule out any LFG-DOP probability model which does not take the LFG Coherence and Uniqueness conditions into account during parsing, as well as Completeness post hoc.

<table>
<thead>
<tr>
<th>#F-structures</th>
<th>=&lt;5</th>
<th>6-9</th>
<th>10-50</th>
<th>51-100</th>
<th>101-1000</th>
<th>1001-10000</th>
<th>10001-100000</th>
<th>&gt;100000</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Sentences</td>
<td>6</td>
<td>2</td>
<td>16</td>
<td>10</td>
<td>18</td>
<td>20</td>
<td>11</td>
<td>17</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 4.6: Explosion of F-structures in a Compacted Grammar

<table>
<thead>
<tr>
<th>#F-structures</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>12</th>
<th>16</th>
<th>18</th>
<th>24</th>
<th>27</th>
<th>32</th>
<th>72</th>
<th>96</th>
<th>Not-tested</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Sentences</td>
<td>4</td>
<td>13</td>
<td>15</td>
<td>3</td>
<td>15</td>
<td>10</td>
<td>14</td>
<td>6</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 4.7: LFG’s Grammaticality Checks act as a Filter on Combinatorial Explosion of F-structures

4.7.2 Enabling Automatic Annotation

Finally, recent and current research (Sadler et al., 2000; Frank et al., 2001) has focussed on further automation. The major bottleneck to the approach of (Van Genabith et al., 1999a,b,c) is the large labour-intensive component, namely manual annotation of the grammar rules. Sadler et al. (2000) show how f-structure annotation of both grammar rules and tree fragments can (to a large extent) be automated.

Again, the idea is a simple one: functional annotations follow systematic patterns. These systematic correspondences between constituent and higher level feature structure representations can be captured in general annotation principles, which are applied to either grammar rules extracted from a treebank or directly to treebanked FS-trees. Sadler et al. (2000) employ annotation principles in order to automate functional structure assignment to flat and ‘noisy’ treebank trees and CFGs extracted from them. Treebanks do not tend to follow highly abstract and general X-bar architectural design principles as are found in LFG and HPSG. The potential benefits of automation are considerable: substantial reduction in development effort, hence savings in time and cost for treebank annotation and grammar development, the ability to tackle larger fragments in a shorter time, a considerable amount of flexibility for switching between different treebank annotation schemes, and a natural approach to robustness.
Sadler et al. (2000) perform an experiment which requires manual inspection, completion and correction of the output produced by the automatic annotation process. They develop an annotation template interpreter with order independent, monotonic interpretation of templates. Consider example (294) from the relative clause section of the grammar:

\[(294) \quad \text{Lhs} \rightarrow \text{Rhs} \mid= [\text{eq}(\text{Lhs}, \text{relcl}: \text{R}), \text{first}(_{-}: \text{T}, \text{Rhs})] @ [\text{R:topic}==\text{T}].\]

This states that if relcl is the category on the Lhs of the rule, and T is the first item on the Rhs of the rule, then automatically add the annotation R:topic==T (i.e. \(\uparrow \text{TOPIC} \) = \(\text{T} \)). Whenever convenient, matching CFG-rules are partially constrained in the Lhs > Rhs component R, as in (295):

\[(295) \quad \text{relcl}: \text{R} \rightarrow [\_] : [.] \mid= \square @ [\text{R:topic}==\text{T}].\]
\[\text{relcl}: \text{R} \rightarrow \text{Rhs} \mid= [\text{mbr}(\text{vp}: \text{V}, \text{Rhs})] @ [\text{R}==\text{V}].\]
\[\text{relcl}: \text{R} \rightarrow [\_] : [\text{Rhs}] \mid= [\text{mbr}(\text{adv}: \text{A}, \text{Rhs})] @ [\text{R:adjunct:element}==\text{A}].\]

(295) states that the initial constituent of a CFG rule for relative clauses is the topic, the head is the sole vp irrespective of its position, and f-structures contributed by non-initial adverbs are collected into an adjunct set. Annotation patterns such as left- or right-headed sequences or co-ordination can make use of parameterized macros involving restricted regular expressions, as in (296):

\[(296) \quad \text{infp}: \text{A} \rightarrow \text{Rhs} \mid= [\text{infix}(\text{Rhs}, \text{conj}([*\text{infp}:-, ?\text{punct}:-], \text{conj}:-, \text{infp}:-], \text{A}, \text{Fd})]) @ \text{Fd}\]

Here (\(? \)) indicates optionality and (\(*, *\) (positive) Kleene star. The template matches coordinate (in the case at hand infinitival) structures with optional punctuation markers and induces a feature structure annotation Fd of the form \([\text{A:pred} = \text{E}, \text{A:conj}:1 = \text{B}, \text{A:conj}:2 = \text{C}, \ldots, \text{A:conj}:n = \text{F}]\).

Given annotation templates such as (294)-(296), Sadler et al. perform an experiment on the same available subset of 100 trees of the \(\text{AP} \) treebank, pre-processed using the structure preserving grammar compaction method reported in (Van Genabith et al., 1999b). A treebank grammar is extracted as in (285), p.153, and the template interpreter compiles the templates over the rules in the treebank grammar. The results obtained are compared against a manually annotated ‘gold standard’ reference grammar. For the experiment reported in (Sadler et al., 2000), 119 templates were constructed against 330 CFG-rules\(^{22}\) resulting in a template/rule ratio of 0.36. This ratio should skew much more in favour of templates as larger fragments are tackled. Automatic annotation generates 1029 annotations, on average about 3.12 annotations per rule. Example output is shown in (297):

\[(297) \quad \text{np}: \text{A} \rightarrow [\text{adj}: \text{B}, \text{punct}:-, \text{adj}: \text{C}, \text{nO}: \text{D}, \text{nO}: \text{E}, \text{nO}: \text{F}]\]
\[\quad @ [[[\text{A}==\text{F}], [\text{A:np_adjunct:1}==\text{B}], [\text{A:np_adjunct:1}==\text{C}; \text{D:headmod}==\text{C}]],\]
\[\quad [\text{F:headmod}==\text{E}, \text{E:headmod}==\text{D}]].\]

---

\(^{22}\)Close inspection of Table 4.5 will show this number to be absent. As stated, the resultant combinatorial explosion in the number of f-structures produced after compaction of the grammar is drastic. Consequently Van Genabith et al. (1999c) reinstate some of the original categories from the uncollapsed grammar as these provide very useful \(\langle c, f \rangle\) linkages and cut down considerably on the number of f-structures generated.
<table>
<thead>
<tr>
<th>Precision</th>
<th>93.38%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recall</td>
<td>91.58%</td>
</tr>
</tbody>
</table>

Table 4.8: Precision & Recall for Automatic Annotation of F-structures using Templates

In terms of precision and recall, the results are shown in Table 4.8. Although good, Sadler et al. (op cit.) report these numbers to be conservative; precision and recall are computed automatically and currently their annotation matcher is not complete. The results are encouraging and indicate that automatic annotation is more often partial than incorrect. The other, perhaps surprising result is that the often held belief that treebanks do not contain useful, consistent information is not is not held up to be true, with the AP treebank, at least. As we pointed out in section 2.1.2, Charniak (1996) makes a similar observation, in that despite common wisdom (or widespread assumption), treebank grammars provide useful resources which outperform other non-word-based (as opposed to tag-based) statistical parsers/grammars on the WSJ treebank.

Despite the small-scale experiments of (Van Genabith et al., 1999a,b,c; Sadler et al., 2000; Frank et al., 2001), the methodology presented is data-driven, semi-automatic and reuses existing resources. Furthermore, the data concerns real language rather than invented, ‘toy’ examples, and the annotation principles express real linguistic generalizations. Automatic annotation holds considerable potential in curtailing development costs and opens up the possibility of tackling the large corpora required as training resources for probabilistic unification grammars and data-driven parsing approaches such as LFG-DOP (Bod & Kaplan 1998; 1999), as well as the translation systems on which they are based, presented in chapter 6.
Chapter 5

Applications of DOP in Translation

We propose here that DOP-based models can be used on a number of levels in the translation process:

- as a translation memory system;
- as a fully fledged MT system using DOP.

We shall discuss the merits of each of these prior to developing our MT proposals based on LFG-DOP in the following chapter.

5.1 DOP as an Exemplar of Translation Memory

One obvious yet novel use for DOP as a model for MT is as an example of Translation Memory (TM) (cf. section 1.2.3), with corpora of linked source and target trees and subtrees. Such a model would act exactly like any other TM system, where new strings are analysed by looking up ‘similar’ examples of previously encountered aligned source and target sentences in a corpus, and suggestions made as to ‘fuzzy’ matches which exceed some predetermined threshold for the user’s selection for post-editing into the final ‘correct’ translation. DOP treebanks would facilitate the alignment process as the syntactic information available would enable the system to better establish links between source and target chunks, rather than leaving the system to try to determine translational equivalents in isolation of such linguistic clues, as is the case with unparsed (or even untagged) corpora.

Like all statistical systems, one of the major attractions of TM is that the larger the corpus becomes, the greater the chance of finding a good translation match. It is generally the case that the more context is available, the better DOP is at resolving such ambiguity. We showed in section 2.1.1 that in this respect DOP acts like PCFGs and unlike HMMs. Most, if not all, TM systems give extra weight to larger chunks, all else being equal, when it comes to constructing new translations. It is, therefore, in the same way unsurprising that DOP’s accuracy in parsing increases when larger chunks are taken into account (Bod 1993b).
5.2 Data-Oriented Translation (DOT)

Notwithstanding the improved production of correspondences between source and target chunks that can be expected by linguistically enhanced corpora such as those provided by DOP, the development of such a system can be viewed as an underachievement of what might be possible if we were to use the full machinery afforded by DOP. All else being equal, we would prefer a dynamic MT system, and in this spirit Poutsma (1998; 2000) has developed such a model—Data-Oriented Translation (DOT). There are two different versions of DOT. We shall look at the details of each in turn.

5.2.1 DOT1

Poutsma’s DOT1 model (1998; 2000) is based on the methodology of Tree-DOP (cf. section 2.1), and relates POS-fragments between two languages (English and Dutch here), with an accompanying probability. DOT1 needs to be parameterized on similar lines to Tree-DOP. Its representations are linked phrase-structure trees. Figure 5.1 shows the complete treebank for the linked sentence pair \( \langle \text{John swims}, \text{Jan zweeft} \rangle \).
These trees are augmented to incorporate semantic information, as a DOT1 treebank links semantically equivalent trees. In order to facilitate this, DOT1 is premised on the Principle of Compositionality of Meaning, which states that the meaning of an expression is a function of its constituent parts and the way they are combined. Poutsma’s definition of semantic equivalence states that two trees $T_1$ and $T_2$ are semantically equivalent iff $T_1$ can be replaced by $T_2$ without loss of meaning. A link between two trees (or subtrees) thus expresses a semantic equivalence between them. A link must exist at the root level, and links may exist at all levels other than at the leaves (cf. Figure 5.1).

In order to form sub-analyses which can be used in translation, Poutsma defines in (298) how source-target DOT fragments are to be linked:

\[(298)\quad \text{“Given a pair of linked trees } \langle T_s, T_t \rangle, \text{ a linked subtree pair of } \langle T_s, T_t \rangle \text{ consists of two connected and linked subgraphs } \langle t_s, t_t \rangle \text{ such that:} \]

1. for every pair of linked nodes $\langle t_s, t_t \rangle$, it holds that:
   (a) both nodes in $\langle t_s, t_t \rangle$ have zero daughter nodes,
   or
   (b) both nodes have all the daughter nodes of the corresponding nodes in $\langle T_s, T_t \rangle$
   and
2. every non-linked node in either $t_s$ or $t_t$ has all the daughter nodes of the corresponding node in $T_s$ ($T_t$),
   and
3. both $t_s$ and $t_t$ consist of more than one node” (Poutsma 2000:24-25).

Poutsma points out that this definition is more restrictive than for Tree-DOP, thus resulting in a fewer (or equal) number of subtrees per tree. For comparison’s sake, a monolingual DOP model for John likes Mary contains 17 subtrees, whereas for Marie plaît à Jean there are 40. These treebanks are shown in Figure 5.2.

We see in Figure 5.3 that a DOT model has just 6 linked fragments for this translation pair. This shows that the issue of the number of fragments is less of a problem in translation than it is in monolingual treebanks.

It should be stressed that both DOT1 and DOT2 produce exactly the same trees—the only difference between the two models is the order in which subtrees are recombined in the target derivation. Note also that given the disjunction in (298-1), only possible subtrees are defined—there will typically be many pairs of linked subtrees per set of linked trees. All linked subtrees produced via (298-1) are semantically equivalent as all semantically equivalent daughter nodes are either removed (298-1a) or retained (298-1b). Nodes which have no semantic equivalence are always retained (298-2).

Having defined the fragmentation process, a derivation for the source language sentence can be arrived at. DOT uses the same composition operation, namely leftmost substitution, as Tree-DOP. The target derivations are then assembled by replacing all subtrees of the source derivation by their linked target subtrees. The probability of target trees that are candidate translations are then calculated using the Tree-DOP probability model (27)–(28), p.36. These need to be adapted slightly to compute the probability of target language subtrees $t_t$. As with Tree-DOP, these are simply their relative frequencies, as in (299):
1: np(john)                                      10: s(np(john), vp(v(_, np(mary))))
2: np(mary)                                     11: s(np(john), vp(v(likes), np(_)))
3: s(np(_, vp(_))                               12: s(np(john), vp(v(likes), np(mary)))
4: s(np(_, vp(v(_, np(_))))                    13: v(likes)
5: s(np(_, vp(v(likes), np(_))))                14: vp(v(_, np(_)))
6: s(np(_, vp(v(likes), np(_))))                15: vp(v(_, np(mary)))
7: s(np(_, vp(v(likes), np(mary))))             16: vp(v(likes), np(_))
8: s(np(john), vp(_))                           17: vp(v(likes), np(mary))
9: s(np(john), vp(v(_, np(_))))                
10: s(np(john), vp(v(_, np(mary))))             
21: s(np(marie), vp(v(_, pp(p(_, np(_))))))    
22: s(np(marie), vp(v(_, pp(p(_, np(jean)))))) 
23: s(np(marie), vp(v(_, pp(p(a), np(_))))))  
24: s(np(marie), vp(v(_, pp(p(a), np(jean))))))
25: s(np(marie), vp(v(plait), pp(_)))           
26: s(np(marie), vp(v(plait), pp(p(_, np(_))))
27: s(np(marie), vp(v(plait), pp(p(_, np(jean))))
28: s(np(marie), vp(v(plait), pp(p(a), np(_))))
29: s(np(marie), vp(v(plait), pp(p(a), np(jean))))
30: v(plait)
31: vp(v(_, pp(_)))
32: vp(v(_, pp(p(_, np(_))))
33: vp(v(_, pp(p(_, np(jean))))
34: vp(v(_, pp(p(a), np(_))))
35: vp(v(_, pp(p(a), np(jean))))
36: vp(v(plait), pp(_))
37: vp(v(plait), pp(p(_, np(_))))
38: vp(v(plait), pp(p(_, np(jean))))
39: vp(v(plait), pp(p(a), np(_)))
40: vp(v(plait), pp(p(a), np(jean))))

Figure 5.2: The monolingual DOP treebanks for the sentences John likes Mary and Marie plait à Jean
Figure 5.3: The complete DOT treebank for the like $\leftrightarrow$ plaie case
\[ P(t_i) = \frac{\#t_i}{\sum_{t' \in \text{root}(t') \cap \text{root}(t_i)} \#t'} \]

The probability of a target derivation is equal to the product of the joint probability of all the individual stochastic events, as in (300):

\[ P(D_t) = P(t_{t_1} \circ \ldots \circ t_{t_n}) = \prod_i P(t_i) \]

Since there are typically many different derivations for the source sentence, there may be as many different translations available. As is the case when DOP is used monolingually, the probability of a translation \( w_t \) is calculated by summing the probabilities of all possible derivations forming the translation, as in (301):

\[ P(w_t) = \sum_{D_t \text{ derives } w_t} P(D_t) \]

These candidate translations can then be evaluated with respect to one another by dividing their probabilities by the sum of the probabilities of all candidate translations, as in (302):

\[ P(\mathcal{T} \mid \mathcal{T}^T = W) = \frac{P(\mathcal{T})}{\sum_{\mathcal{T}' \models W} P(\mathcal{T}')} \]

\( \mathcal{T}^T = W \) means that \( \mathcal{T} \) is a translation of a source word string \( W \). This process is similar to sampling valid LFG-DOP representations (cf. (221), p.116) from an LFG-DOP corpus. Poutsma shows that the most probable translation can be computed using Monte-Carlo disambiguation,\(^1\) and exemplifies this using sentence idioms. Poutsma’s system is premised on the Principle of Compositionality of Translation, namely that two strings are considered to be translations if and only if they have been constructed from parts which are each others translations, as well as the Principle of Compositionality of Meaning. Given this, it is rather unfortunate that Poutsma selects sentence idioms to exemplify his system, as the meaning of sentences containing such phenomena are not constructed compositionally. One also has to doubt whether such a linguistic phenomenon is ideally suited to automatic translation, despite the fact that it is quite common to see developers of MT systems attempting to show how their systems cope with idiomatic material.

Poutsma (2000:47-48) tests to see which of the translations in (303) are more likely for \textit{the woman kicked the bucket}:

(303) a. De vrouw schopte de emmer (literal translation)

b. De vrouw ging de pijp uit (idiomatic translation)

The small experiment centres on the use of all subtree fragments of the linked trees in (304):

\(^1\) Recall that Goodman (1998) shows that Monte-Carlo disambiguation has exponential time-complexity. We described Simaan’s (1999) improvements in the efficiency of DOP models in section 4.6.
Poutsma found that the idiomatic reading (303b) was selected as the most probable translation 90% of the time. This reading is the more likely due to DOT’s preference for generating a translation with the largest possible corpus fragments. For example, one possible derivation of (303b) is (305):

That is, almost the complete topmost tree fragment in (304) can be used to construct (303b). In contrast, the largest tree fragment that can be used to construct (303a) from the bottom tree pair in (304) is that involved in the derivation shown in (306):
We can illustrate further how DOT1 would work with our simple example sentences John swims ⇔ Jan zwemt and Peter laughs ⇔ Piet lacht. Given suitable DOT treebanks for these sentences (as in Figure 5.1), we can now evaluate the four strings in (234)—John swims, Peter laughs, John laughs, Peter swims—together with their Dutch equivalents. Given that the number of subtrees for John swims in Figure 2.1 is identical and isomorphic to the trees in Figure 5.1, the DOT1 probability of each English sentence in (234), p.123 is identical to its DOP probability. Given also that except for the leaves themselves, the Dutch fragments are identical to the English fragments, these will have the same probabilities, and so will the translations. That is, the probabilities of the translations are as in (307):

\[
\begin{align*}
(307) \ a. \ & \ P(\text{John swims} \leftrightarrow \text{Jan zwemt}) = 7/24 \\
& \ b. \quad P(\text{Peter laughs} \leftrightarrow \text{Piet lacht}) = 7/24 \\
& \ c. \quad P(\text{John laughs} \leftrightarrow \text{Jan lacht}) = 5/24 \\
& \ d. \quad P(\text{Peter swims} \leftrightarrow \text{Piet zwemt}) = 5/24
\end{align*}
\]

Once again, the presence of complete parse trees for (307a,b) cause their probabilities to exceed those of the two other possible sentence pairs. Of course, \( P(\text{John swims} ^T \leftrightarrow \text{Jan zwemt}) = 1 \) given a vacuous application of (302). Indeed, as there are no other translations possible for any of the sentences in (307), each translation given formula (302) has a probability of 1, but this will not always be the case: it is merely an artefact of the simple examples used here.

5.2.2 Some Limitations of DOT1

DOP-based models of translation may be considered to be an instance of example-based approach to MT (cf. Somers, 1999). The problems mentioned in section 1.2.3 for EBMT, namely boundary definition and boundary friction, are not as problematic as in ‘pure’ EBMT systems, which store translations as unannotated strings. Nevertheless, the presence of the syntactic categories in DOP trees does not always help ensure that well-formed structures are derived. To give an example, in attempting to translate the sentence The PC business effects changes in its marketing strategy for its European operations, an EBMT system may have the fragments in (308) in its memory on which to base its judgements:

169
(308) a. The PC business effects are wide-ranging in the Asian economy.

b. British Telecom changes radically in its marketing strategy for the next century.

This might cause it to produce the incorrect translation with *changes* as the verb and *effects* as a noun, rather than the other way round, assuming no other suitable chunks are available. DOP too may err here; it is possible to insert the VP headed by *changes* into the tree (309):

![Tree diagram](image)

In this case, we end up with a wrong (yet valid with respect to the corpus) analysis for this sentence. Depending on the corpus, however, it is reasonable to expect that the probability of other correct derivations may exceed that of this incorrect analysis. This shortcoming would be overcome by augmenting a DOP treebank with LFG f-structures, as LFG’s unification element would prevent such a derivation with a plural SUBJ NP and a singular VP.

In addition, DOT1 may not produce the correct translation when faced with certain example sentences, yet when DOT2 and LFG-DOT are used this problem can be overcome. For example, the *like* $\rightarrow$ *plaire* (relation changing) case cannot be dealt with in the DOT1 model. The bag of linked subtrees for the *like* $\rightarrow$ *plaire* case is shown in Figure 5.3, p.166. It is interesting that if we attempt to translate *John likes Mary*, unless there is some prior occurrence of *Jean* as object, or *Marie* as subject, DOT may actually prefer the wrong translation *Jean plait à Marie*. If we have a treebank built up from *Jean embrasse Marie* and *Sarah plait à Bill*, then the string *Jean plait à Marie* is about 1.25 times more likely than the correct alternative in the French language model. The reason for this is DOP’s preference for *Jean* as subject, given the occurrence of tree (310) already in its treebank:

![Tree diagram](image)

The *like* $\rightarrow$ *plaire* case shows up a problem with the DOT1 composition operator. As soon as derivation (311) of the source sentence is arrived at, the desired linking of the English SUBJ with the French prepositional OBJ, and that of the English OBJ with the French SUBJ, are overridden by the composition operation of DOT1:

![Tree diagram](image)
In this case the wrong translation of (311) is derived, as in (312):

Poutsma suggests that this may be overcome by redefining the composition operation of DOT1 to operate on pairs of trees, rather than on single trees. This improvement is incorporated in DOT2, which is discussed further in section 5.2.3.

Poutsma himself (2000:377.) points out further flaws in the DOT1 model. A problem at the level of transfer is that source derivations may be built which cannot be transferred into a target translation. An example would be where non-terminal frontier nodes are labelled with different categories than the root nodes of other trees they are to be combined with. Again, a redefinition of the DOT1 composition operation to ensure that each source derivation has a valid, unique target derivation is the solution proposed. Given that the new composition operation is defined on inextricably linked tree pairs, the target tree is built up synchronously with the source tree, so that the definitions of source and target are merged.

A problem with the probability model of DOT1 as defined in (299)-(301), p.164 is that only the target probabilities are factored into the calculation of the probability of a translation. Therefore potentially important source language information is being jettisoned. DOT2 factors both the probability of the source and target sentences into its translation probability model.

Furthermore, DOP’s statistical model also gives a ‘level of correctness’ figure to alternative translations. This is useful (though must be treated with caution, as it may rank wrong translations above correct alternatives, cf. (333) below) in cases where the default translation in LFG-MT (and in many other systems) cannot be suppressed when the specific translation is required (cf. commit suicide, (112-115), p.81). Let us assume a DOP treebank built from the French sentences in (313):

(313)  a. Jean commet un crime ↔ John commits a crime
       b. Le suicide est tragique ↔ Suicide is tragic
       c. Marie se suicide ↔ Mary commits suicide

If we now analyse the ill-formed string Jean commet le suicide, we observe that it is preferred in the French language model about half as much again as the correct alternative Jean se suicide. There are several
reasons for this: the preference for _Jean_ as subject of _commettre_, the co-occurrence of _le_ and _suicide_, plus the fact that _commettre_ is followed by an NP consisting of a Det + N sequence. Again, we note an adverse consequence of DOT1’s preference for translations built from larger fragments. Note also that producing more than one translation for a string is not possible with LTAG-MT (Abeillé et al., 1990), for instance, so in this case we assume that the more likely French string will be proposed as the wrong, final translation.

It should be stressed that these results are obtained with the same number of instances of each verb—in a larger corpus _commettre_ would surely greatly outnumber instances of _se suicider_. Nor are these results unexpected. As an example, in the _LOB Corpus_ (cf. chapter 4, note 18), there are 66 instances of _commit_ (including its morphological variants), only 4 of which have _suicide_ as its object, out of the 15 occurrences of _suicide_ as an NP. Consequently, even for this small sample, we can see that 94% of these examples need to be translated compositionally (by _commettre_ + NP), while only the _commit suicide_ examples require a specific rule to apply (i.e. _se suicider_). In the on-line Canadian Hansards covering 1986-1993, there are just 106 instances of _se suicider_ (including its morphological variants). There will, of course, be many thousands of instances of _commettre_. Given occurrences of _suicide_ as an NP in French corpora, it is not an unreasonable hypothesis to expect that the wrong translations such as _John commits suicide_ $\rightarrow$ *_Jean commet le suicide_ in (115), p.82, will be much more probable than those derived via the specific rule. We examine the _commit suicide_ example further in the section 5.2.6 (cf. (334), p.179, and resultant discussion).

These are by no means isolated cases of difficult translation problems. Way _et al._ (1997) produce a categorization of a number of ‘hard’ cases of translation containing complex insertions and deletions, such as:

1. ‘Schimmel’ cases (also classified as cases of ‘Conflational Divergence’ (Dorr 1993:258f.), or ‘1-to-N lexical gaps’ (Lindop & Tsujii, 1991), from the well-known example where the German noun _Schimmel_ translates as the English governor _horse_ plus a complete AP modifier containing _white_.

2. Relation changing verbs, such as the _like $\leftrightarrow$ plaire_ case, Figure 5.3, p.166.

3. ‘Shoehorn’ cases, such as (17), p.29, _savoir $\leftrightarrow$ know how_, where an additional piece of structure needs to be ‘shoehorned in’ by the target grammars around an already existing piece of target structure.

4. Headswitching cases, such as (43a), p.48, or _Ich arbeite gerne $\leftrightarrow$ I like working_ (cf. chapter 1, note 2), where what in English is realized as a main verb is expressed in German by means of an adverbial modifier, _gerne_.

These category mismatches, lexical holes, insertions, and deletions can be described in similar terms to those used in _Candide_ (Brown _et al._, 1990; 1992a)—_fertility and distortion_. Whichever description one chooses, the mechanics of DOT1 remain the same whatever pair of languages one is translating between. Nevertheless, we expect it to do better between languages with similar word-order (cf. (345) and (348), p.184ff., for instance), in a similar way to transfer, which prefers ‘like’ languages, whereas the interlingual approach is often quoted as being better for dissimilar ones.

In sum, it would appear that the adherence to left-most substitution in the target given a priori left-most substitution in the source is too strictly linked to linear order of words, so that, as soon as this deviates to any significant extent even between similar languages, DOT1 has a huge bias in favour of the incorrect translation. Even if the correct, non-compositional translation is achievable in such circumstances, it is likely to be so outranked by other wrong alternatives that it will be dismissed, unless all possible translations are maintained for later scrutiny by the user.

5.2.3 DOT2: An Improved Model of Translation

DOT2 is identical to DOT1 except for the definition of its composition operation and an improved probabilistic model which takes the probability of both source and target sentences into account when calculating the probability of a translation. The new composition operation continues to use leftmost substitution, but now pairs of trees are composed:

“The composition of the linked tree pair \( \langle t_s, t_t \rangle \) and \( \langle u_s, u_t \rangle \), written as \( \langle t_s, t_t \rangle \circ \langle u_s, u_t \rangle \), is defined iff the label of the leftmost non-terminal linked frontier node and the label of its linked counterpart are identical to the labels of the root nodes of \( \langle u_s, u_t \rangle \). If this composition is defined, it yields a copy of \( \langle t_s, t_t \rangle \), in which a copy of \( u_s \) has been substituted on \( t_s \)'s leftmost non-terminal linked frontier node, and a copy of \( u_t \) has been substituted on the node's linked counterpart” (Poutsma 2000:39).

This new definition ensures, among other things, that relation-changing cases such as like \( \leftarrow \rightarrow \) plaire are handled correctly: instead of the wrong derivation (312) of the source (311), we now obtain the correct derivation (314):

\[
(314) \quad \begin{array}{c}
S \\
\text{NP} & \circ & \text{NP} & \circ & \text{NP} \\
V & PP & Marie & Jean & S \\
plait & P & NP & Jean & V \\
& & â & & plait \\
& & & & a \\
& & & & Marie
\end{array}
\]

The calculation of subtrees in DOT2 takes pairs of source-target subtrees into account. The probability of a source-target subtree pair \( \langle t_s, t_t \rangle \) is simply its relative frequency, as in (315):

\[
(315) \quad P(t_s, t_t) = \frac{\#(t_s, t_t)}{\sum_{\langle u_s, u_t \rangle: res = \langle u_s, u_t \rangle \Rightarrow res = \langle t_s, t_t \rangle} \#(u_s, u_t)}
\]

The probability of DOT2 derivations is altered to factor in subtree pairs, as in (316):

\[
(316) \quad P(\langle t_{s1}, t_{t1} \rangle \circ \ldots \circ \langle t_{sn}, t_{tn} \rangle) = \prod_i P(\langle t_{si}, t_{ti} \rangle)
\]

The probability of a translation \( \omega_s \Rightarrow \omega_t \) is calculated by summing the probabilities of all possible derivations forming the translation, as in (317):

173
(317) \[ P(w_x, w_t) = \sum_{D(x, t)} P(D(w_x, w_t)) \]

Poutsma shows that subtree linkages such as (318) can be rewritten in rule format:

![Diagram](image)

(318) \( \langle S, S \rangle \rightarrow (\langle \text{John, likes, } NP \rangle, \langle NP, \text{plait, a, Jean} \rangle) \)

This is illustrated in (319):

(319) can combine with any rule having the root pair \(\langle NP, NP \rangle\). Given that we are now dealing with what amount to CF-rules, algorithms can be used which ensure that the parse-time complexity is polynomial with respect to the input length (e.g. Younger, 1967; Earley, 1970). Disambiguation, on the other hand, cannot be guaranteed in polynomial time, as there may be exponentially many derivations of sentences and translations. We stated earlier that Bod & Scha (1997:158-162) try to use Viterbi optimization to find the most probable derivation rather than the most probable parse, but results degrade significantly. Given this, Poutsma (2000:46) prefers to use Monte-Carlo methods to randomly select target derivations from the set of all candidates. However, Hoogweg (2000) found that the Monte-Carlo algorithm described in (Bod, 1998:46-49) is not guaranteed to find a unique random derivation in some cases. For instance, consider the set of elementary trees in (320):

(320) \[
\begin{array}{c}
S \\
A \quad B \\
a \quad b \\
\end{array} \quad \begin{array}{c}
A \\
B \\
c \\
a \\
\end{array} \quad \begin{array}{c}
A \\
B \\
a \\
b \\
c \\
\end{array}
\]

If we now parse the string \(abc\), then randomly selecting trees with root nodes A and B in turn and deleting other trees with the same respective root nodes will not eliminate any of the elementary trees as these span different parts of the input sentence, so both analyses in (321) remain:

(321) \[
\begin{array}{c}
S \\
A \quad B \\
a \quad b \\
\end{array} \quad \begin{array}{c}
S \\
A \quad B \\
a \quad b \\
c \\
\end{array}
\]

Given this, Poutsma chooses to adapt Hoogweg’s top-down, breadth-first method rather than use bottom-up, breadth-first processing to allow for pairs of trees in his experiments with DOT2.
After the small experiments of (303)-(306), Poutsma tested DOT2 with 266 sentences from the English-German section of the Verbmobil corpus. We stated in section 2.2 that DOP1 cannot cope with unknown words. As DOT is based on DOP1, Poutsma ensured that all the words in the testset were contained in the training set. Poutsma correctly observes that evaluating MT output is difficult, and so selects a two-stage evaluation: (i) whether the output is an exact, alternative, partially correct or wrong translation of the source; and (ii) what is the largest translation part (LTP) of the output translation, in terms of how many continuous words in the translation are correct.

For English-German, 24% of the translations were deemed exactly correct or alternative translations, and 38% were partially correct. For German-English, 23% of the translations were deemed exactly correct or alternative translations, and 56% were partially correct. Increasing the depth of fragments increased these success rates somewhat. The LTP results for sentences of less than 9 words was about 50%, and from 10%-17% for longer strings, depending on the depth of subtrees used.

Poutsma notes that these results can be explained by many factors. The corpus is small, notwithstanding the fact that corpus preparation took 200 hours (approx. 45 minutes per sentence) using dedicated software on a fast machine. Furthermore, the quality of the English translations in the corpus is poor in places—Poutsma considers that about 5% of the translations are wrong or ungrammatical, which improves the perception of his results somewhat. Additionally, agreement caused a number of ungrammatical, yet almost correct sentences, such as those in (322):

\[
\begin{align*}
\text{Source:} & \quad \text{I will book the trains.} \\
\text{(322) a. Translation:} & \quad \text{Ich werden die Züge reservieren.} \\
& \quad \text{Correct:} & \quad \text{Ich werde die Züge reservieren.} \\
\text{Source:} & \quad \text{It is a private organizer.} \\
\text{Translation:} & \quad \text{Es ist eine privat Veranstalter.} \\
& \quad \text{Correct:} & \quad \text{Es ist ein privat Veranstalter.}
\end{align*}
\]

That is, in the first example DOT2 has produced a translation with a singular subject and a plural verb, whilst in the second example, we see a feminine article instead of a masculine article. It must be stated that of these examples quoted by Poutsma (2000:37), neither source sentence could be considered absolutely correct English. Nevertheless, the German translations obtained can easily be corrected by post-editors and so have obvious merit as candidate translations. Finally, there were a number of translations with almost exactly the same probabilities, but different categories. In such cases, re-running experiments caused different alternative translations to be selected, which caused a significant drop in results, particularly with such a small testset of 40 sentences. Of course, it is clear that there are very few large, good quality bilingual treebanks available on which to test DOP-based translation systems. We addressed these concerns in section 4.7 with respect to LFG-DOP treebanks.

**Translations in Context**

We noted in section 2.1.1 an advantage of DOP over PCFGs, namely DOP’s ability to capture collocations which occur outside the context of individual rewrite rules, as DOP is not constrained by an underlying
grammar. As Bod & Kaplan (1999:2) note:

“some constructions of natural language have dependencies ... that cannot be accounted for by
the free interaction of smallest independent rules .... the rule formalisms of most linguistic theories
embody the smallest/independent bias so strongly that they make it difficult to characterize the
special properties of larger units of language”.

DOP, on the other hand, shows no such bias, and correctly captures such collocations in a natural way.
For example, we showed in section 3.3.1 that LFG-MT (Kaplan et al., 1989) was not able to handle certain
translation cases such as *vieux ludothèque* $\rightarrow$ *old toy library* (172b), p.98, where the adjunct set contains
multiple members. Assuming a representative corpus, we can expect to find examples like (323) in DOT
models:

\[
\begin{array}{c}
\text{NP} \\
\text{Adj} \quad \text{N} \\
toy \quad \text{library} \\
\text{NP}
\end{array}
\]

(323)

This shows the special relationship between *toy* and *library*, as well as the fact that together their translation
is *ludothèque*. In the case of *old toy library* $\rightarrow$ *vieux ludothèque* (180a), p.100, we would have (324):

\[
\begin{array}{c}
\text{NP} \\
\text{AP} \quad \text{Adj} \\
\text{old} \quad \text{toy} \quad \text{vieux} \\
\text{NP}
\end{array}
\]

(324)

This shows the default translation of *old* as *vieux*, implying that the remainder of the two trees are translations
of each other. Once ( *old,vieux*) are linked, they can be deleted so that any linked adjectival pair can be
inserted at the *Adj* nodes. There is no reason to suppose that DOT will not translate these examples correctly
given the existence of such links.

DOP-based translation should maintain this advantage over any rule-based system. For instance, we can
contrast DOP translation models such as DOT2 (and LFG-DOT), which can provide parts of translations in
context, with the *Eurotra* translation formalisms *CAT* and *EF*, which we described in section 1.3.3 cannot.
In linking problematic parts of source-target fragments, (at least) some pairs of linked fragments correspond
to translation rules. We should, therefore, be able to deal with ‘hard’ cases, and combinations of exceptions,
assuming that instances of the specific translations do appear in the corpus.

176
We showed in section 1.3.3 that CAT was not able to handle cases such as (17c), *On sait le faire* $\leftrightarrow$ *How to do it is known*. In contrast, DOT is like *Mimo* in that its models may correctly prefer specific translations for sentences like (17c) over those incorrect ones produced by default rules, although this is not always the case (cf. (332) and (333) below). It can be seen quite straightforwardly that a tree fragment (generalized in (325) with infinitives) will be produced which corresponds exactly to the *Mimo* rule (24), p.32:

\[(325)\]

```
  VP          VP
  \|--|       \|--|
  \|  \       \|  \       savoir          know
  V   S       V   how
```

The V-node dominating *savoir* is not linked to the V-node dominating *know* as these are not translationally equivalent. Rather, the fact that *savoir* is the only unlinked element on the source side and *know* and *how* are unlinked on the target side indicates a more complex, implicit linkage akin to what one might expect in a transfer rule.

Let us adapt the examples in (17), p.29, slightly to those in (326) in order to cut down on the number of fragments which are irrelevant to the question at hand:

\[(326)\]  

a. Jean sait le faire $\leftrightarrow$ John knows how to do it.  
b. On mange bien $\leftrightarrow$ One eats well.

Now (17c), namely *On sait le faire*, can only be translated compositionally as *One knows how to do it*, which if not exactly wrong, is certainly not as acceptable as *How to do it is known* or *It is known how to do it*. We shall see later in the chapter that in a similar manner to DOT’s treatment of complex cases such as headswitching and relation-changing cases, these ‘hard’ cases are not dealt with in as compositional a manner as one would like. In some cases of complex transfer, as here, there are many fewer links between translationally equivalent nodes as there are in simpler cases which require just default translation. Taking *On le fait maintenant* $\leftrightarrow$ *It is done now* in (17b) as an example, the complete set of DOT fragments are shown in (327):
(327) signifies that these fragments are suitable only for exact matches or for cases of adverb insertion. In order to achieve the correct translation \textit{On sait le faire $\leftrightarrow$ How to do it is known} in (17c), we would have to add fragments built from exactly this sentence pair, or another sentence in which this was a complete sub-structure. The default compositional translation \textit{One knows how to do it} would still be derivable, and given DOT's preference for translations formed from bigger linked trees, this may be preferred. Deriving a DOT treebank from (17c) and (326) gives the results in (328):

\begin{align*}
\text{(328) a. } & \quad P(\text{On sait le faire} \leftrightarrow \text{One knows how to do it}) = \frac{39}{1064} = 0.036654 \\
\text{b. } & \quad P(\text{On sait le faire} \leftrightarrow \text{It is known how to do it}) = \frac{126}{1064} = 0.11842
\end{align*}

Using formula (302) we can rank these translations with respect to one another, as in (329):

\begin{align*}
\text{(329) a. } & \quad P(\text{On sait le faire} \overset{T}{=} \text{One knows how to do it}) = 39/165 = 0.2363 \\
\text{b. } & \quad P(\text{On sait le faire} \overset{T}{=} \text{It is known how to do it}) = 126/165 = 0.7637
\end{align*}

Let us now increase the number of instances of the translation pairs and associated linked fragments from (326) on the assumption that these sorts of sentences are more likely than (17c). Thus we are attempting to show the impact of more frequent trees compared to more specific trees on the probabilities of translations. In each subsequent iteration, the number of derivations does not change, just their probabilities. With 5 sentences, (17c) plus twice those in (326), we now get the results in (330):

\begin{align*}
\text{(330) a. } & \quad P(\text{On sait le faire} \leftrightarrow \text{One knows how to do it}) = \frac{70}{1666} = 0.042017 \\
\text{b. } & \quad P(\text{On sait le faire} \leftrightarrow \text{It is known how to do it}) = \frac{105}{1666} = 0.063025
\end{align*}

The ranking of the respective translations against each other is shown in (331):
(331) \[ P(\text{On sait le faire} \ T = \text{One knows how to do it}) = 70/175 = 0.4 \]
\[ P(\text{On sait le faire} \ T = \text{It is known how to do it}) = 105/175 = 0.6 \]

With 7 sentences, (17c) plus three times those in (326), we now get the results in (332):

(332) \[ P(\text{On sait le faire} \longleftrightarrow \text{One knows how to do it}) = 43.2/980 = 0.044082 \]
\[ P(\text{On sait le faire} \longleftrightarrow \text{It is known how to do it}) = 42/980 = 0.042857 \]

The respective translations are ranked against each other in (333):

(333) \[ P(\text{On sait le faire} \ T = \text{One knows how to do it}) = 43.2/85.2 = 0.507 \]
\[ P(\text{On sait le faire} \ T = \text{It is known how to do it}) = 42/85.2 = 0.493 \]

Thus it can be seen that the preferred translation can quite quickly be undermined by the less preferred compositional alternative if the corpora are made ‘more representative’. Despite the fact that the exact translation is available in (332) and (333), it is not preferred over the compositional alternative. Of course we are merely hypothesizing that the data in (326) are ‘more likely’ than (17c): when confronted with real data this hypothesis may be rejected. In practice, given the closeness of the result in (333), one would expect that both translations would be presented to the user for selection. Indeed, even where there is greater disparity between the relative probabilities of candidate translations, this possibility remains open to the system designer.

Thus the issue of representativeness of corpora needs to be always kept in mind. When dealing with the sort of difficult translational phenomena we are interested in here, this concern is almost impossible to maintain with any empirical rigour. Nevertheless, we feel that the experiments reported here give a flavour of how DOP-based translation systems perform.

**Testing Translation Candidates in DOT2**

In section 5.2.2 we discussed the commit suicide case (cf. also (112)-(115), p.81), and hypothesized that the wrong, compositional translation tends to be more probable than the specific, non-compositional alternative, given that we can expect many more instances of commettre in a French corpus than instances of se suicider. However, increasing the number of instances of commettre does not mean that all specific translations are overruled by their default, compositional alternatives. We produced a DOT treebank containing all the linked fragments from the sentences in (334):
Table 5.1: Number of Fragments in Two Monolingual Treebanks

<table>
<thead>
<tr>
<th>Sent</th>
<th>English Fragments</th>
<th>French Fragments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>S</td>
<td>NP</td>
</tr>
<tr>
<td>S1</td>
<td>15</td>
<td>2</td>
</tr>
<tr>
<td>S2</td>
<td>22</td>
<td>5</td>
</tr>
<tr>
<td>S3</td>
<td>22</td>
<td>5</td>
</tr>
<tr>
<td>S4</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>S5</td>
<td>14</td>
<td>3</td>
</tr>
<tr>
<td>S6</td>
<td>22</td>
<td>5</td>
</tr>
<tr>
<td>S7</td>
<td>22</td>
<td>5</td>
</tr>
<tr>
<td>S8</td>
<td>22</td>
<td>5</td>
</tr>
<tr>
<td>S9</td>
<td>22</td>
<td>5</td>
</tr>
<tr>
<td>S10</td>
<td>22</td>
<td>5</td>
</tr>
<tr>
<td>Totals</td>
<td>189</td>
<td>41</td>
</tr>
</tbody>
</table>

(334) a. S1: Le suicide est tragique ↔ Suicide is tragic.

b. S2: Jean commet le crime ↔ John commits the crime.

c. S3: Jean commet le meurtre ↔ John commits the murder.

d. S4: Jean dort ↔ John sleeps.

e. S5: Marie se suicide ↔ Marie commits suicide.

f. S6: Marie commet un attentat ↔ Mary commits an attack.

g. S7: Marie commet la faute ↔ Mary commits the mistake.

h. S8: Pierre commet un arbitre ↔ Peter nominates an arbitrator.

i. S9: Pierre commet une erreur ↔ Peter commits an error.

j. S10: Pierre commet une injustice ↔ Peter commits an injustice.

Here there are seven instances of commettrire (six of which translate as commit) as opposed to only one instance of se suicider. The breakdown in terms of fragments for the English and French sentences in (334) are shown in Table 5.1. Note that the major differences in Table 5.1 concern S1 (14 more fragments in the French string) and S5 (15 more fragments in the English string). The French monolingual DOP treebank built from the sentences in (334) contains 337 fragments. The equivalent English treebank, at 338 fragments, contains one more fragment owing to the more complex \( v_P(v(work),N_P(N(work))) \) compared to its French equivalent \( v_P(v(se suicide)) \), with a slight trade-off in the other direction owing to the omission of a determiner in Suicide is tragic compared to its French equivalent in (334a). The maximum number of candidate fragments that can participate in a linked DOT treebank is, therefore, 352 = 29 = 323 fragments. That is, add the 14 extra French fragments from S1 to the larger English number, and subtract the sum of the differences (14 + 15 = 29). In fact, there are only 321 linked fragments, as no linking at the verb level is possible in S5 (cf. 337 below), and one of the English NPs in S1 cannot be linked to its French counterpart. The NPs from S1 in (335) can be linked:
However, given that the two *suicide* nouns are not linked, we cannot produce the link between the nouns in (336):

French *suicide* can never occur without an article, so it is the combination of *le* and *suicide* which translates as *suicide* in English. If we had *the suicide* in English, we could draw the links between both determiners and both nouns, but this is not possible in (336). Similarly, we can make the link between the VPs in S5 explicit in (337):

However, there is nothing to link \( v(\text{se suicide}) \) to. It is the combination of *commit* and *suicide* that translates as *se suicider*, so a more fine-grained compositional translation is impossible here. Once again, we see that a DOT treebank contains fewer fragments than the monolingual treebanks from which it is derived.\(^3\)

In the monolingual French DOP treebank, *Marie se suicide* is preferred about 2.6 times over the compositional alternative *Marie commet le suicide*, as shown in (338):

\[
(338) \quad \begin{align*}
\text{a. } & P(\text{Marie se suicide}) = 0.006443 \\
\text{b. } & P(\text{Marie commet le suicide}) = 0.002472
\end{align*}
\]

If these were the output translations, then ranking them against one another would favour *Marie se suicide* with about 72% versus *Mary commet le suicide* with about 28%. In the DOT2 treebank produced from the English and French sentences in (334), the specific translation is preferred even more than in the French monolingual DOP treebank. We set out to test the weight of the specific over the compositional translation for the sentences in (339):

\(^3\)The same will be seen for LFG-DOT treebanks, as each DOT fragment is linked one to one with its corresponding f-structure fragment.
Figure 5.4: DOT Derivations for *John commits suicide* ↔ *Jean se suicide*

(339) a. John commits suicide ↔ Jean se suicide

b. Mary commits suicide ↔ Marie se suicide

c. John commits suicide ↔ *Jean commet le suicide*

d. Mary commits suicide ↔ *Marie commet le suicide*

Translation (339a) can be built using the three derivations in Figure 5.4. (339b) has the additional derivation of the full trees for this sentence pair. The probabilities of (339a-b) are shown in (340):

(340) a. \( P(\text{John commits suicide} \leftrightarrow \text{Jean se suicide}) = 0.000705 \) \((\approx \frac{1}{1419})\)

b. \( P(\text{Mary commits suicide} \leftrightarrow \text{Marie se suicide}) = 0.006229 \) \((\approx \frac{1}{161})\)

For each of the translations in (339c-d) there are 7 derivations with total probability 0.000501 \((\approx \frac{1}{1998})\). Now we can rank each translation with respect to the other in (341):

(341) a. \( P(\text{John commits suicide} \overset{T}{=} \text{Jean se suicide}) = 705/1206 = 0.5846 \)

b. \( P(\text{John commits suicide} \overset{T}{=} \text{Jean commet le suicide}) = 501/1206 = 0.4154 \)

c. \( P(\text{Mary commits suicide} \overset{T}{=} \text{Marie se suicide}) = 6229/6750 = 0.923 \)

d. \( P(\text{Mary commits suicide} \overset{T}{=} \text{Marie commet le suicide}) = 521/6750 = 0.077 \)

Therefore we can see that for *John commits suicide*, the correct, specific translation is about 1.4 times more likely than the wrong, default, compositional translation, whereas for *Mary commits suicide* the specific translation is preferred about 12 times more than the default translation. In contrast to the examples in (328)-(333), we see in (341) the dominance of the exact linked translation pair over the alternative translation. In (328)-(333) we were adding exact copies of the sentences in (326) so the build up of similar fragments cumulatively outperforms the single instance of tree fragments from (17c), p.29. In (334) on the other hand, in an attempt to keep the corpus ‘representative’, the slightly different (but similar) sentences cause the build up of *commettre* fragments to have an ultimately lesser effect than the fragments built from (334a).
The presence of the exact translation (334e) (i.e. S5) is of course insufficient to explain the preference for the specific translation for *John commits suicide*: despite the presence of six *commit* \leftrightarrow *commettre* examples in (334) compared to only the single instance of *commits suicide* \leftrightarrow *se suicide* example, the specific translation is nonetheless preferred.

We do not know empirically how many more times we can expect to see *commit* \leftrightarrow *commettre* compared to *commits suicide* \leftrightarrow *se suicide*, but it is clear from the results in (341) that a ratio of 6:1 is insufficient to achieve a bias in favour of the wrong, compositional translation. Running a new experiment with a treebank built from 9 instances of each translation pair (334a-d) and (334f-j) and just the one instance of (334e), making a total of 46 sentences in all, produces the results in (342):

\[(342)\quad \begin{align*}
a. \quad & P(\text{John commits suicide} \overset{T}{=} \text{Jean se suicide}) = 132/635 = 0.208 \\
b. \quad & P(\text{John commits suicide} \overset{T}{=} \text{Jean commet le suicide}) = 503/635 = 0.792 \\
c. \quad & P(\text{Mary commits suicide} \overset{T}{=} \text{Marie se suicide}) = 1206/1758 = 0.686 \\
d. \quad & P(\text{Mary commits suicide} \overset{T}{=} \text{Marie commet le suicide}) = 552/1758 = 0.314
\end{align*}\]

Now, with 30 instances of *commit* \leftrightarrow *commettre* and only the one *commits suicide* \leftrightarrow *se suicide* example, we see that the default, compositional translation for *John commits suicide* is now preferred by about 3.8 times, but the presence of the exact translation (334e) maintains the preference for the specific translation for *Mary commits suicide* by about 2.2 times. Consequently we can see that it will take many more instances of *commit* \leftrightarrow *commettre* before the specific translation for *Mary commits suicide* is outranked by the wrong, compositional alternative.

The issue of representativeness of test corpora is actually a very tricky one. The results in (342) are skewed slightly by the fact that in increasing the number of instances of the sentences in (334), there are now only 11 instances of *Mary* \leftrightarrow *Marie* compared to 15 instances of *John* \leftrightarrow *Jean*, which results in $P(\text{John commits suicide} \overset{T}{=} \text{Jean commet le suicide})$ being about 1.15 times higher than $P(\text{Mary commits suicide} \overset{T}{=} \text{Marie commet le suicide})$. Furthermore, adding fragments which differ even with respect to whether the NP OBJ of *commettre* is definite or indefinite skews the probabilities of the translations slightly. If large, high-quality translational corpora were available on which to test our hypotheses, we may hope that this issue may disappear.

**DOT2 and some ‘hard’ Translation Cases**

Let us now evaluate how the DOT2 model copes with some of the headswitching examples from section 3.2. We are interested in discovering to what extent the presence of headswitching examples influences translation cases which require compositional (default) translation, and vice versa. We will base our discussion on the data in (343):

183
(343)  a. DE: Johannes schwimmt gerne ↔ EN: John likes to swim.
       b. DE: Josef läuft zufällig ↔ EN: Joseph happens to run.

Presupposing the derivation of a monolingual treebank constructed from the German examples in (343), with two different NPs, verbs and adverbs, 2^3 sentences are possible and can receive analyses with associated probabilities with respect to that DOP corpus. However, only four of these possible sentences can receive translations in a DOT corpus, namely the examples in (343) as well as those in (344), by simple substitution of the alternate NPs into the respective SUBJ slots:

(344)  a. DE: Josef schwimmt gerne ↔ EN: Joseph likes to swim.
       b. DE: Johannes läuft zufällig ↔ EN: John happens to run.

The other four sentence pairs in (345) cannot be handled at all in a DOT treebank built from the strings in (343):

(345)  a. DE: Johannes läuft gerne ↔ EN: John likes to run.
       b. DE: Josef schwimmt zufällig ↔ EN: Joseph happens to swim.
       c. DE: Josef läuft gerne ↔ EN: Joseph likes to run.
       d. DE: Johannes schwimmt zufällig ↔ EN: John happens to swim.

This is due to the fact that the linked VP pairs are not broken down any further than at the root level. The contrast can be seen by examining the schwimmt gerne VP and its constituent DOP-fragments in Figure 5.5 and (346), which contains the single linked VP DOT pair:

(346)

We cannot draw a link between schwimmt and swim in (346) as they are not translationally equivalent. We cannot, therefore, draw links between the fragments in (347), as we might otherwise wish to do, in order to describe the basic translation relations in (343):

184
In such circumstances, the only way that the sentence pairs in (345) can be handled is if there is some prior linked pair läuft gerne ←→ likes to run as well as a prior instance of the linked pair schwimmt zufällig ←→ happens to swim. This is because these linked VP pairs are handled non-compositionally in DOT2 between German and English, but the monolingual VPs are treated compositionally in DOP. Contrast this situation with a DOT treebank designed to translate these 8 German strings into Dutch. Our starting point could be the German strings in (343) with their Dutch translations, as in (348):

\[(348)\]
\[
\begin{align*}
\text{a. DE: Johannes schwimmt gerne} & \leftrightarrow \text{NL: Johan zwemt graag}, \\
\text{b. DE: Josef läuft zufällig} & \leftrightarrow \text{NL: Josef loopt toevalig}.
\end{align*}
\]

Given this simple transfer example, a DOT treebank would resemble much more closely the monolingual DOP treebanks from which it is derived for the respective sentences in (348), as every constituent part of the German strings corresponds exactly to a constituent part of the Dutch strings. In the DOT treebank these links are made explicit. When we have a headswitching case, however, it is apparent that both DOT models would translate the sentences correctly iff prior examples of linked headswitching VPs exist in the treebank. For instance, let us add the sentence pairs with all other resultant linked fragments in (349) to those in (343):

\[(349)\]
\[
\begin{align*}
\text{a. DE: Peter läuft gerne} & \leftrightarrow \text{EN: Peter likes to run}, \\
\text{b. DE: Markus schwimmt zufällig} & \leftrightarrow \text{EN: Mark happens to swim}.
\end{align*}
\]

This now allows all 8 German strings in (343)-(345) to be correctly translated into English by substitution of the relevant SUBJ NPs in this trivial corpus. They would receive extremely low probabilities with respect
to the corpus in the normal case as they are built with a minimal degree of compositionality (substitution of SUBJ NP). As these examples only ever occur rarely, the chances of DOT2 managing to translate these in practice becomes significantly lower than might otherwise be expected, as we require not only the presence of the adverb, but also its occurrence to be correlated exactly with the verb in question for translation to succeed. The chances of such a co-occurrence are small, but we suppose that using larger corpora will enable the translation to be correctly rendered.

5.3 Outstanding Issues

It is clear that DOT2 is an improvement on DOT1. Nevertheless, a number of questions remain, notwithstanding the improved composition operation and probability model of DOT2, which enable the correct linkages in (319), p.174, and similar examples. In DOP and LFG-DOP treebanks, it is clear that fragment definition allows the generation of these respective treebanks completely automatically. In DOT it is not clear that this is possible any more: how can a process be defined so that verb-pairs such as *like* ↔ *plaire* are treated differently from ‘normal’ transitive verbs? Poutsma’s notation for treating fragment pairs like (318) as rules such as (319) implies, as in other rule-based systems, a transfer component consisting of both default and specific rules. The main benefits of the statistical approach, namely the ability to maintain consistency in large-scale systems and interpreting uncertainty in an objective manner are lost somewhat in the DOT2 approach—we are back at a rule-based system, albeit with associated probabilities, and admittedly only for the examples in the corpus.

There is a whole class of relation-changing verbs for which a generalized notation ought to enable us to treat the class as a whole. For example, only the tree-pair in (350) out of all the linked fragments for the *like* ↔ *plaire* in Figure 5.3 can possibly play a role in the translation of *Peter likes Susan* ↔ *Suzanne plait à Pierre*.

![Diagram](image)

(350)

No other S-fragments in the DOT treebank can be used, unless there are prior examples of *Peter likes* or *likes Susan* in the treebank. One might expect that derivations such as (351) may be possible:

186
There will, however, be no links at all between \emph{like} and \emph{plaire} as they are not semantically equivalent. What \emph{is} semantically equivalent is the tree-pair in (350). Thus no substitution of (a morphological variant of) \emph{plaire} can occur at the V node in (351). Furthermore, not even one S-fragment in Figure 5.3, p.166, can be used in the translation of \emph{John misses Mary} $\iff$ \emph{Marie manque à Jean}, despite the fact that the translation relation between \emph{like} $\iff$ \emph{plaire} is identical in every respect to \emph{miss} $\iff$ \emph{manquer}. There is, therefore, an amount of redundancy in the DOT2 model of translation.

Given this, it should be apparent that just as with the headswitching examples (343)-(345), very few trees can play a role in the formation of translation involving a relation-changing verb in DOT2, even if there are many instances of such verb-pairs in the corpus. The translation pairs will, therefore, also have very low probabilities with respect to the corpus as their compositional derivations are very few. Maybe this is a good thing: compared to DOT1 we can no longer get wrong translations only—as we are writing what are in effect transfer rules to ensure the correct translation, correct translations are obtained along with some possible wrong alternatives. Unlike transfer rules, however, we do not have the problem (as we do in LFG-MT) of ensuring that the default, compositional alternative is suppressed when a specific rule fires. Furthermore, it may be the case that the DOT2 way of treating relation-changing verbs compared with transitive verbs gives some psychological plausibility to the treebank. There will be many instances of the latter in any representative corpus, so there will be many different ways of deriving translations for this class of verb. In such cases, all fragments in a source Tree-DOP treebank are linked one to one with their equivalent structures in the target treebank. On the other hand, we can expect relatively few instances of relation-changing verbs in our corpora, which is mirrored by the fact that very few derivations are possible with such verbs, so correct translations which are theoretically possible may not be so achievable in practice.

A related problem concerns the pruning of the search space which is usually required in order to facilitate practical testing of DOP-based models. We have seen that this is typically achieved by a combination of limiting the depth of fragments playing a role in derivations of analyses as well as restricting the number of terminals allowed in tree fragments. Given that complex examples such as \emph{On le fait maintenant} $\iff$ \emph{It is done now} in (327) undergo rather less fragmentation in DOT translation models than their monolingual counterparts, restricting the number of terminals may not be possible with such cases. If this is so, manual intervention to exclude such instances from the automatic pruning process may be required, otherwise such examples will need to be flagged in some way to prevent their undergoing pruning. Of course, if they are subjected to the pruning process, their claim to be linked pairs of translationally equivalent chunks is undermined. Furthermore, whilst restricting the number of terminals in a monolingual model is completely straightforward, it is far from that in any model of translation: given that we are dealing with \emph{pairs} of
structures, examples such as Figure 5.3 show that it is not the case that X words in one language map onto X words in another. Therefore, placing a restriction of (say) two terminals per monolingual tree may result in further ‘ill-formed’ fragment pairs in DOT models (cf. section 6.3.6. for discussion of lexicalization in LFG-DOT models).

One further problem is that Poutsma’s DOT models cannot distinguish ill-formed from well-formed input. For example, both DOT models of translation would permit the derivations in (352):

That is, with no stipulation on subject-verb agreement, it is perfectly legitimate in DOP-based models to combine a singular subject with a plural verb and end up with a well-formed object. In DOT, we end up with a translation which is well-formed given the corpus. As soon as grammatical information is available, such a combination process would be impossible. In the next chapter, we shall show that our LFG-DOT models use Discard to relax grammaticality constraints such as these to allow for processing of such ill-formed input. This improves the robustness of LFG-MT models, but importantly such translation pairs will be regarded as ungrammatical with respect to the corpus given their derivation via Discard; in DOT models, we have no such distinction.

One of the advantages of DOP-based models over LFG-MT is that they have a built in robustness. Indeed, this characteristic was one of the main reasons behind the amalgamation of LFG and DOP to produce LFG-DOP. DOT, being based on DOP1, cannot deal with unknown words. It would not, however, be too onerous to add the techniques of DOP2 and subsequent models (cf. section 2.2) to both DOT models to enable some treatment of unknown words. It would seem that the DOT2 transfer rules considerably reduce the amount of ill-formed input that can be dealt with. In DOT1, verbs like plaîre are treated in exactly the same way as unproblematic cases; in DOT2, they get ‘special’ treatment.

DOT2 performs better on well-formed input than DOT1. It remains an open question as to which system would work better on (varying degrees of) ill-formed input. DOT2 gets the like plaîre hard case right, whereas DOT1 does not. In going from DOT1 to DOT2, Poutsma has traded some of DOT1’s robustness in dealing with ill-formed strings to improve the translation quality when treating well-formed sentences. Ultimately, of course, any approach to NLP which is based purely on trees cannot hope to deal successfully with all the problems that language throws up. For this reason, we develop in the next chapter models of translation based on LFG-DOP, which are able to handle phenomena requiring greater than context-free generative power.
Chapter 6

LFG-DOT: A New, Hybrid Model of Translation

6.1 Introduction

At this point in the thesis, we have demonstrated the advantages and disadvantages of using certain types of formalism as the basis for a translation system:

- LFG: a constraint-based notation;
- CAT: a ‘rule-to-rule’ (or rather ‘representation’ to ‘representation’) notation;
- DOT: a probabilistic tree-based system.

While LFG’s $\tau$-equations are in the main able to link exactly those source-target elements which are translations of each other, leading to elegant translation models such as that of Kaplan et al. (1989), we showed in chapter 3 a number of cases where this machinery is unable to cope with a subset of translation cases, in particular embedded headswitching examples.

Way et al., (1997:324) state that “CAT ... [has] certain features which would appear in anyone’s first approximation to a translator notation, based as they are on the local subtree of a linguistic representation”. Nevertheless, the very nature of the local tree restriction in CAT is what causes problems for this system, especially when confronted with combinations of exceptional translational phenomena.

Neither LFG-MT nor CAT factor probabilistic models into any process in their MT systems. We consider that structures with associated probabilities are desirable: they may allow probabilistic ranking of the structures (and translations) produced, and furthermore allow pruning, should this be required. In contrast to CAT and LFG-MT, the DOT models use the statistical language processing model DOP as the basis for their translation systems. Nevertheless, the DOT1 model cannot always explicitly relate parts of the source language structure to the corresponding, correct parts in the target structure, so fails to translate correctly where source and target strings differ with respect to word order. DOT2 was developed as a consequence of
these failings, and indeed manages to overcome them in most cases. However, this is compromised by a lesser amount of compositionality in the translation process. Given the small number of fragments playing a role in the derivation of some translations involving complex phenomena, almost the exact linked sentence pair may need to be present in order for a translation to be possible. Furthermore, any such translations produced have extremely small probabilities with respect to the corpus. Finally, of course, translation systems which are based purely on PS-trees will ultimately not be able to handle certain linguistic phenomena.

We now propose the use of LFG-DOP (Bod & Kaplan, 1998; 1999) as the basis for an innovative MT system. We present here a number of possible translation models: as a preamble, we hypothesize a model which combines both DOP and LFG, but which does not use LFG-DOP per se. While such a model is of (at least theoretical) interest as an example of a combination of statistical and constraint-based processing, given that the two methods are not combined into a coherent, embedded approach, it remains somewhat peripheral to the central core of this thesis. The first two models proper propose the use of LFG’s τ-equations to relate translation fragments between languages, the second in combination with γ, which we use to refer to the function that links DOT source and target subtree fragments. Using τ-equations overcomes some of the problems of the DOT1 translation model. In addition, DOP adds robustness to LFG-MT, both with respect to dealing with ill-formed input, and to dealing with well-formed input not covered by the treebank.

Given that the τ mapping cannot always produce the desired translation, LFG-DOT Model 3 jettisons τ-equations and relies on γ to express the translation relation. In section 6.2.3, we compare this model to DOT2 and CAT, and argue that as LFG-DOT3 is conceptually different, it avoids some of the problems of these other tree-based systems. Nevertheless, like DOT2, LFG-DOT3 suffers from limited compositionality. Therefore, we advocate the use of a restricted form of Discard in an ‘extended transfer’ phase to generalize the translation relation in a desirable manner, resulting in our final model, LFG-DOT4. We shall see that this model describes the translation relation exactly as required, and furthermore overcomes the problems of LFG-MT, CAT and DOT models of translation.

Following the presentation of the various system architectures, we present probability models for each LFG-DOT system. These are all based on the relative frequency estimator of a fragment in a treebank. Other researchers have pointed out errors or insufficiencies with such estimators, and we present these accounts here. Finally, we comment on the impact of Discard generated fragments on our probability models, and discuss how these might be best accounted for in any probability model.

### 6.2 Possible Models for LFG-DOT

As we have just outlined, there seem to be a number of possibilities as to how LFG-DOP MT might work. We detail these further in sections 6.2.1. through 6.2.4. At the outset, however, we note the following possible combination of DOP and LFG, Model 0, which nonetheless does not use LFG-DOP itself. This could be organized as follows:

1. Start with a source Tree-DOP treebank which produces monolingual PS-trees in the usual way.
2. Once these structures have been built, look up the PRED values of the leaves on the tree in an appropriate LFG lexicon, and other syntactic features in an LFG grammar.

3. Use the LFG semantic forms to filter out some of the structures built on the grounds of subcategorization violations (LFG completeness and coherence checks). This presupposes that complete f-structures have been constructed in step (2).

4. Look up lexical and structural τ-equations in the lexicon and grammar respectively.

5. Use these to produce target f-structures.


Such a model would suffer from the DOP and LFG-MT problems described previously, but adding linguistic knowledge from LFG is likely to lead to an improvement in translation quality over pure DOP-based methods. We showed in chapter 5 that both DOT models permit the derivations in (352) for the linked pair ('John swim, Jan zwemmen'), so that an ill-formed object is produced by combining legitimate fragments. Nevertheless, the resultant ‘translation’ is considered grammatical given the database. As soon as grammatical information is available via LFG f-structures, such a combination process is impossible unless the Uniqueness grammaticality check is violated. Knowledge about the grammaticality of strings (and structures) is important to guide the probability models. In chapter 4 and here, we advocate the splitting of fragments into well-formed and ill-formed bags.

A more sophisticated version of Model 0 would be to have a bilingual DOT-treebank to produce linked <source,target> pairs of trees with associated probabilities. Steps 4-6 from Model 0 could then be used to derive translations, with the further interim step being the use of the probabilities derived to rank the f-structures produced. Such an alternative would continue to suffer from the DOP and LFG-DOT problems mentioned, but the probabilistic ranking of f-structures may help to overcome the problem of LFG-MT where several target f-structures are output with no associated ranking (in terms of probabilities) attached. The ranking of f-structures is discussed further in section 6.2.2.

Despite the advantages noted of Model 0 over DOT and ‘pure’ LFG-based models of translation, we will demonstrate that models based on LFG-DOP are able to produce the correct translations where DOT and LFG-based models may not. Furthermore, LFG-DOT models are more robust, given the ability of the Discard operation to relax constraints where appropriate. We will also present LFG-DOT4, which overcomes the DOT2 problem of limited compositionality. We now concentrate on instantiations of four specific models which use the machinery of LFG-DOP to varying degrees.

6.2.1 LFG-DOT Model 1: Translation via τ

This is a simple, linear model. Given a source language LFG-DOP treebank, the model builds a target f-structure \( f' \) from a source c-structure \( c \) and f-structure \( f \), the mapping between them LFG-DOP-\( \phi \), and
the LFG translation equations $\tau$. From this target f-structure $f'$, a target string is generated via a target language LFG-DOP model,\footnote{Note that LFG-DOP-$\phi'$ is not a function: one only has to think of free word order languages to see immediately that one f-structure can represent many different strings.} as in (353):

$$
\begin{array}{c}
\text{LFG-DOP-$\phi$} \\
\downarrow \tau \\
\text{LFG-DOP-$\phi'$}
\end{array}
\begin{array}{c}
c \\
\longrightarrow \scriptstyle{\rightarrow} \\
\longrightarrow \downarrow \tau \\
d' \leftarrow \scriptstyle{\leftarrow} \\
\end{array}
\begin{array}{c}
f \\
\end{array}
$$

The different components needed are:

- a source language LFG-DOP model;
- the $\tau$ mapping;
- a target language LFG-DOP model.

We give probability models for LFG-DOT1 in section 6.3.1.\footnote{The probability models in section 6.3.1 below in effect assume two separate (but related) models of LFG-DOT1. The first given probability model describes the main LFG-DOT1 model proposed here. However, it is clear that a variant of this is possible where the target string is generated using well-known standard LFG generation algorithms (cf. Wedekind 1988, 1999; Kohl 1992; Wedekind & Kaplan, 1996), rather than by a target language LFG-DOP model. This has the effect of altering the system architecture in (353) to that in (354):}

In order to see more precisely how an LFG-DOT Model 1 might work, let us first take the simple example sentences in (355):

(355) a. John swims $\iff$ Jan zwemt.

b. Peter laughs $\iff$ Piet lacht.

LFG-DOT1 presumes a monolingual source corpus; if we assume this to be English, then this corpus would be as in Figure 4.1, p.114, with similar additions for Peter laughs. For the four sentences in (234), p.123, which are possible given this treebank, LFG-DOP would produce the structures in (356) as input into the translation phase:
We now need to consult the $\tau$-equations in (357):

\[ \begin{align*}
\text{swim} & \\
(\tau \uparrow \text{PRED}) = \text{zwemmen} & (\tau \uparrow \text{PRED}) = \text{lachen} \\
(\tau \uparrow \text{SUBJ}) = (\tau \uparrow \text{SUBJ}) & (\tau \uparrow \text{SUBJ}) = (\tau \uparrow \text{SUBJ}) \\
\text{John:} & \\
(\tau \uparrow \text{PRED}) = \text{‘Jan’} & (\tau \uparrow \text{PRED}) = \text{‘Piet’}
\end{align*} \]

These $\tau$-equations enable the target f-structures in (358) to be built:

\[ \begin{align*}
\text{SUBJ} & \quad \text{PRED ‘Jan’} \\
& \quad \text{NUM SG} & \quad \text{SUBJ} & \quad \text{PRED ‘Piet’} \\
& \quad \text{TENSE PRES} & \quad \text{PRED ‘zwemmen((↑SUBJ))’} & \quad \text{TENSE PRES} \\
& \quad \text{PRED ‘lachen((↑SUBJ))’} & \quad \text{TENSE PRES} & \quad \text{TENSE PRES}
\end{align*} \]

Target strings can now be generated from the f-structures in (358) via LFG-DOP-$\varphi$ in a target language LFG-DOP model.

The main advantage of LFG-DOT1 compared to LFG-MT is the added robustness. LFG-DOT1 contains two monolingual LFG-DOP language models, so Discard can be run on both source and target sides. This means that LFG-DOT1 can cope with ill-formed or previously unseen input which LFG-MT would not be able to handle at all. Suppose that John swim is encountered as the source string. If we assume that the top left linked $\langle c, f \rangle$ pair in (356) exists in our LFG-DOT treebank, then the $\text{TENSE} = \text{PRES}$ constraint can be relaxed by Discard to allow the derivation of the $\langle c, f \rangle$ pair in (359):
Another \((c,f)\) pair would be built via Discard where the \(\text{NUM} = \text{SG}\) feature is removed. Ignoring this for the time being, the f-structure in (359) would be input into the translation phase, which in LFG-DOT1 is quite simply the LFG-MT \(\tau\) function. Taking (359) as input, the \(\tau\)-equations in (357) would build the target f-structure in (360) assuming Dutch to be the target language:

\[
\begin{array}{c}
\text{SUBJ} \\
\text{PRED} \\
'\text{Jan}'
\end{array}
\]

\[
\begin{array}{c}
\text{NUM} \\
\text{SG}
\end{array}
\]

\[
\begin{array}{c}
\text{PRED} \\
'\text{zwemmen}((\uparrow\text{SUBJ}))'
\end{array}
\]

The target language LFG-DOT model would link (360) to the appropriate c-structure tree, as in (361):

\[
\begin{array}{c}
\text{NP} \\
\text{John}
\end{array}
\]

\[
\begin{array}{c}
\text{VP} \\
\text{swim}
\end{array}
\]

\[
\begin{array}{c}
\text{SUBJ} \\
\text{PRED} \\
'\text{Jan}'
\end{array}
\]

\[
\begin{array}{c}
\text{NUM} \\
\text{SG}
\end{array}
\]

\[
\begin{array}{c}
\text{PRED} \\
'\text{zwemmen}((\uparrow\text{SUBJ}))'
\end{array}
\]

The ‘translation’ of the ill-formed string \(\text{John swim}\) would therefore be \(\text{Jan zwemmen}\). This can easily be post-edited into a well-formed target string. We discuss the impact of Discard on the translation models in section 6.3.6.

With respect to more complex examples, the advantage of this model over DOT1 is the availability of the explicit \(\tau\)-equations to link source-target correspondences. Recall that DOT1 was unable to handle the \(\text{like} \leftrightarrow \text{plaire}\) case correctly (cf. Figure 5.3, p.166, and resultant discussion). LFG-DOT1 uses \(\tau\) to express the translation relation, so the LFG-MT solution (82), p.73, to this relation-changing case can be availed of quite straightforwardly.

However, as might be expected, most of the LFG-MT problems are imported into this model. Specifically, using LFG \(\tau\)-equations ensures the derivation of the correct target f-structure in most cases, along with some possible wrong alternatives via the default rules. For the \textit{commit suicide} \(\leftrightarrow \text{se suicider}\) example, LFG-DOT1 produces the source f-structure (362) from a number of different LFG-DOP derivations:
We give three such derivations in Figure 6.1.

Assuming an LFG-DOP corpus derived from the English strings in (334), p.179, for instance, only the f-structure (362) will be created for *John commits suicide*, as all derivations result in the same analysis. This will have a certain probability with respect to the corpus, but as it is the only f-structure created we can ignore this here. As there are no competitors, it goes forward as the sole input to the translation process. Nevertheless, we know that LFG-MT cannot suppress the wrong, default translation where an alternative translation via a specific lexical entry is produced. We can, therefore, expect the two target language f-structures in (363) to be built:

The first f-structure in (363) is derived via the specific τ-equations in (116), p.82, whereas the second f-structure in (363) is derived via the default τ-equations in (113), p.81, the regular entry for *commit*. Unless a target language LFG-DOP model is added, both candidate f-structures would be submitted to the target string generation stage.\(^3\) This example is an illustration of one of the remaining problems for MT discussed in section 1.3, namely ambiguity—*commit* is not ambiguous in English, but the LFG treatment of translation using τ-equations renders it ambiguous in translation. Accordingly, the French target grammar will be confronted with the two f-structures in (363), one which should produce an acceptable French string and the other which does not, so should be filtered out. In subsequent LFG-DOT models, if such structures are obtained, they will appear with associated probabilities, enabling them to be ranked and pruned, if required. LFG-MT outputs them both with no preferences attached.

\(^3\)A linguistic expert would be required to select the better translation. Alternatively, the user could step in prior to the generation of the string and select the best candidate f-structure as input into this process.
Figure 6.1: Sample LFG-DOP derivations for John commits suicide
Summary

LFG-DOT1 can solve some of the problems of DOT1, and has certain advantages over DOT2. *Discard* improves the robustness of LFG-MT. However, in using the $\tau$ mapping for transfer, like LFG-MT it does not get some of the ‘hard’ translation cases right. We know that DOT2 can handle these cases correctly (cf. relation-changing (82)f., p.73; *commit suicide*, (336)f., p.181; and headswitching (346)f., p.184). It seems sensible, therefore, to introduce the $\gamma$ relation into our translation models in order to link source and target subtree fragments. This is the LFG-DOT2 model of translation.

6.2.2 LFG-DOT Model 2: Translation via $\tau$ and $\gamma$

This model requires integrated bilingual LFG-DOP corpora, where each node $n$ in a source c-structure tree $c$ is related both to its corresponding f-structure fragment $f$ (via LFG-DOP-$\phi$) and its corresponding c-structure node $n'$ in a target c-structure tree $c'$ (via $\gamma$). In addition, each f-structure fragment $s$ in a source f-structure $f$ is related to its corresponding language fragment $s'$ in a target f-structure $f'$, via $\tau$, as in (364):

\[ \begin{array}{c}
\text{LFG-DOP-$\phi$} \\
\gamma \quad \text{c} \quad \downarrow \quad f \\
\text{LFG-DOP-$\phi'$} \\
\tau \quad c' \quad \downarrow \quad f'
\end{array} \]

Model 2 contains explicit links between both surface constituents and f-structure units in both languages, whereas LFG-DOT1 relates the languages just at the level of f-structure (via $\tau$). Here $\gamma$ is the DOT2 model of translation outlined in section 5.2.3. Consequently, LFG-DOT2 is a more complex model, necessitating:

- a source language LFG-DOP model;
- the $\gamma$ mapping (i.e. the DOT2 model of translation);
- a target language LFG-DOP model;
- a probabilistic transfer component.

With respect to this latter component, the question of integration of the $\gamma$-probabilities with the $\tau$ mapping to derive a translation, either via the most probable combination of linked LFG-DOP fragments, or by using DOP techniques purely on f-structures, is addressed below. Probability models for LFG-DOT2 are given in section 6.3.2.
As we did with Model 1, let us now proceed through some translation examples using the more complex machinery of Model 2. The LFG-DOT2 treebank (no Discard) for John swims $\iff$ Jan zwemt is given in Figure 6.2.\footnote{Treebanks of this type would serve as input into LFG-DOT Models 2-4. We have already pointed out that LFG-DOT1 treebanks consist merely of source c-structure trees and their corresponding f-structures. Target f-structures are created by means of $\tau$-equations.} As well as the source and target $\phi$-links shown, we assume DOT-links between translationally equivalent parts of the source and target c-structures. Given the triviality of the corpus chosen, the necessary links should be evident from the previous discussion, so we have omitted these for reasons of clarity. Adding the LFG-DOT2 corpus for (355b), p.192 (which would mirror Figure 6.2 except for the leaves on the c-structure trees and f-structure PRED values) to Figure 6.2 enables us to handle the two new translations in (365):

(365) a. John laughs $\iff$ Jan lacht.
   
   b. Peter swims $\iff$ Piet zwemt.

As with the DOT examples, the probabilities of all four translations are the same as those assigned under DOP. The fact that we have to factor in f-structure fragments makes no difference here given the trivial corpus: source c-structures are combined in an identical fashion to Tree-DOP (cf. Figure 2.4, p.42, and resultant discussion), and as in DOT, the target c-structures are built simultaneously. Given that source and target fragments are $(c, \text{LFG-DOP-$\phi,f$})$ triples, the source and target f-structures are built up in a corresponding synchronous fashion. Depending on which LFG-DOP probability model is chosen, the process by which the structures are built differs: whether we use just the Tree-DOP Root category matching condition (model M1), LFG’s Uniqueness condition (M2), Coherence check (M3) or Completeness check (M4) will affect neither the actual structures built nor their probabilities for the simple translation pairs studied here.

Figure 6.3 shows a schematic picture of how LFG-DOT2 copes with the like $\leftrightarrow$ plaire case (cf. Figure 5.3, p.166). Starting at the top-left, the source string John likes Mary is analysed by an English LFG-DOP component to produce the topmost f-structure, which is input into the transfer process. Using the LFG $\tau$-equations in (82), p.73, will give us the lower target f-structure to be input into the generation phase. If more than one f-structure is produced as a candidate, then these can be automatically ranked using the notion of $\tau$-support in (366) below. The set of f-structure linked-pairs can be produced with or without Discard depending on the level of robustness (generalization of fragments) required.\footnote{Note that if we were to omit Discard from a model of MT, we would have an LFG-DOT model which would suffer from lack of robustness similar to the LFG-MT models described in chapter 3.} Alternatively, if only a small set of structures are produced then these can be subjected to manual evaluation. In addition, the bottommost c-structure tree will be produced via $\gamma$ from the source c-structure,\footnote{It could also be derived from the target f-structure using standard LFG generation techniques, as in the variant to LFG-DOT1 in (354), but we do not consider this possibility further here.} and the string trivially derived.

The main reason for the addition of the $\gamma$ operation in LFG-DOT2 is that the correct translation is not always achievable via $\tau$-equations (cf. chapter 3). Having already shown that DOT2 is able to produce correct translations (Poutsma 1998, 2000; cf. also chapter 5), harnessing $\gamma$ to the $\tau$-equations in a probabilistic transfer component should provide better results than using $\tau$-equations alone in Model 1, as we have more
Figure 6.2: The complete LFG-DOT2 treebank (no Discard) for John swims ↔ Jan zweent
Figure 6.3: LFG-DOT2: Translating using Different Levels of Linguistic Representation
information to bring to bear in resolving potential conflicts.⁷

Maintaining an \( f \) to \( f' \) translation engine in addition to \( \gamma \) increases the likelihood of achieving the correct translation—even if this is not proposed as the most probable translation via \( \tau \), given that this function will only ever produce very few translation candidates, we can guarantee that it is suggested as one of a small set of candidate translations. These can be compared to the best translation generated by \( \gamma \) and the highest ranking overall translation selected as output.

Alternatively, if the \( \gamma \) and \( \tau \) translation relations are permitted to operate on the same probability space, then the applicability of \( \tau \) needs to be restricted in terms of its impact on the probabilities of candidate translations. We use Good-Turing to limit the effect of Discard on the probabilities, and propose the application of the same technique here. In this way the two functions (\( \gamma \) and \( \tau \)) are combined, but \( \tau \) is limited to a smaller section of the probability space. Therefore the \( \gamma \)-links have priority over the \( \tau \)-equations, but they work in harness. Whichever approach is taken, the production of translations via \( \gamma \) and \( \tau \) might be viewed in the same light as in the multi-engine systems described in section 1.2.4., where such translation candidates are used to mutually confirm the best translation.

Another reason for restricting the scope of \( \tau \) is that the target f-structure built may not be acceptable to the target language model,⁸ either because it is missing as an exemplar from that model (i.e. it is unable to be deconstructed via LFG-DOP-\( \sigma' \) into a c-structure), or because given the target model, it is ill-formed. In this respect the LFG-DOT models presented so far may be open to the criticism that they fall foul of the subset problem (cf. section 1.3.2), another of the remaining problems for MT. We consider that this is unlikely to happen when the f-structure is the only candidate: if the \( \tau \)-equations are correctly specified, then a well-formed target f-structure will ensue. Of course, in the case of headswhitting cases at least, the correct translation might remain unobtainable, but if the constraints have been solved correctly, then some linguistic object will be derivable. In cases where two target f-structures are put forward (as in (363), for instance), we expect the target language LFG-DOP model to prune undesired f-structures proposed by default \( \tau \)-equations.

---

⁷Both LFG-DOT1 and LFG-DOT2 models could be extended to cope with LFG \( \sigma \)-structures (cf. Butt 1994), with the further addition of a mapping \( \sigma' \) to relate target f- and \( \sigma \)-structures (i.e. \( f' \) and \( \sigma' \)), to bring still more information to bear on the translation process. This would tie in nicely with the extensions to DOP to cope with semantics which have already been developed (Bonnema et al. 1997). Kaplan & Wedekind (1993) introduced the notion of restriction as an attempt to solve some of the problematic translation cases LFG-MT faced, especially headswhitting examples, with a further translation phase taking place at c-structure in addition to the regular \( \tau \)-equations used to relate languages at the level of f-structure. We showed in chapter 3 that this additional device fails to cope with all such problems, and introduces further complications. Given the additional machinery required to enable one’s system to cope with this added complexity, especially when regarded in the light of the continued inability to cope with certain ‘hard’ translation cases, one wonders whether incorporating a further level of processing is worthwhile. The general point remains that nothing in the proposed design of any LFG-DOT model precludes additional levels of processing, although we consider that LFG-DOT4 in particular enables a sufficient statement of the translation relation in its own right.

⁸Instead of hypothesizing a target language LFG-DOP model, we might assume instead that as well as information about potential target c-structures being to hand (via \( \gamma \)), we might resort once more to the generation of the target string via standard LFG generation algorithms (cf. section 3.4). Wedekind (1988:735) notes that f-structure constraints should be dropped “until an output sentence can be derived. The control of such a dropping procedure is... dependent on the application domain”. He exemplifies this claim with respect to MT, adding (opus cit., p.737) that “this dropping of constraints could be controlled in such a way that a maximal similarity with respect to the stylistic features of the f-structure of the source language sentence has to be ensured. What ‘maximal similarity with respect to the stylistic features’ means again presupposes some empirical work”.

201
where instead the one derived via the specific τ-equations is the correct, intended output. For example, in (277) we showed that the PP in the sentence *John waited for Mary* may be interpreted either as an oblique *for-OBJ*, or as an adjunct. Corpus analysis may show that the argument reading is to be preferred over the modifier alternative, in which case the target language LFG-DOP model will reflect this and propose the oblique option as the correct analysis. Alternatively, it is not difficult to envisage a procedure whereby lexical solutions are preferred over structural alternatives.

**Using DOP Techniques on a ‘Treebank’ of F-structures**

We consider here a variation of LFG-DOT2 which uses DOP-techniques to rank alternative target f-structures. DOP is a general methodology—the form of the objects does not matter at all. Candidate f-structures are pruned so that only the highest ranked goes forward to the generation stage.9 We require a definition for decomposition, which is based on the notion of ‘τ-support’ in (366) (cf. (249), p.129, the LFG-DOP notion of *support*):

(366) A source f-structure unit *s* τ-supports a target f-structure unit *t* if and only if

- *t = τ(s)*, or
- *t* is the value of some grammatical-function attribute in τ(*s*), or
- *t* is the value of some non-grammatical-function attribute in an f-structure *v*, where *v* is τ-supported by *s* and *t* is not τ-linked”.

τ-links are quite simply elements of two τ-supported f-structures linked via a τ-equation. Given the definition of τ-support in (366), we are able to specify f-structure linked fragments as in (367):

(367) Given an LFG representation ⟨*s*, τ, *t*⟩, where *s* τ-supports *t*, ⟨*s’*, τ’, *t’*⟩ is an LFG-DOT linked f-structure fragment if τ’ is the sub-correspondence of τ that relates the f-structure units in *s’* to the f-structure units in *t’*.

In this version of LFG-DOT2, recomposition of f-structure fragments is performed using unification of f-structure fragments, as with LFG and LFG-DOP.

As an example of the fragmentation process, let us take the f-structure pair for (355a), p.192, shown in (368):

---

9This may not lead to the selection of the best translation. We saw earlier in (115) (the *commit suicide* example, p.82) that a monolingual DOP model favours the wrong, compositional string *John commit le suicide over the specific, correct translation, but that DOT1 achieves the opposite but desired result, namely preference for the specific translation using *se suicider* in (341), p.182. Only large-scale experimentation will show whether this pruning of f-structures leads to successful results.
The set of additional linked f-structure fragments derivable from (368) is shown in (369):

\[
\begin{align*}
&\text{SUBJ} & \text{PRED} & \text{NUM} & \text{SG} \\
&\text{PRED} & \text{swim}(\uparrow\text{SUBJ})' \\
&TENSE & \text{PRES} \\
\end{align*}
\]

The topmost pair may be derived by deleting the content of the SUBJ f-structures, and the second pair by deleting the PRED and associated TENSE attribute-value pairs. TENSE cannot remain on its own as it is supported (φ-linked) by the outer PRED. As such, the combination of source outer PRED and TENSE are τ-linked to the target outer PRED and TENSE. Further linked f-structure fragments are derivable from (368) by using Discard to selectively delete non-grammatical-function attributes from linked f-structure pairs. Taking (368) and (369) as input, the linked f-structure fragments derivable via Discard are those in Figure 6.4. The first three pairs are formed from (368) by deleting SUBJ:NUM:SG, TENSE=PRES and both respectively, and the bottom two are formed by deleting TENSE=PRES from the first pair in (369) and SUBJ:NUM:SG from the second pair, respectively.

The complete set of linked f-structure fragments derivable from Figure 6.2 is shown in Figure 6.5 (no Discard). The original f-structures from Figure 6.2 appear in unitalicized form, while those fragments derived via definition (367) are in italics. Of course, as we have seen there are no duplicate target f-structures produced in the John swims ⇔ Jan zwent example. In the commit suicide example, however, we saw the two candidate target f-structures in (363) built. Supposing we have fragmented all f-structures for the relevant corpus (from (334), p.179, say) into τ-linked pairs as here, we can produce all derivations for this pair of f-structures only—we do not need to derive their probabilities with respect to the corpus of f-structures, so we do not need to calculate their frequencies relative to the corpus as a whole. Rather, what is needed is to calculate their probabilities with respect to each other, to see in effect what their relative worth as translations is. LFG-DOT has already produced them as candidate target f-structures in the translation process via its τ-equations—all that remains is for them to be ranked against each other so that their likely effectiveness as input to generation can be evaluated. 10 This is exactly what was seen earlier when evaluating candidate translations in DOT, where the τ operator was used (cf. (341), p.182, for instance) to indicate the ranking of each translation with respect to the others.

---

10 A further, simpler, yet in all probability less successful ranking model would be to calculate the relative frequencies of functional descriptions. Such a model would, however, not draw a link between (say) \( f_a(PRED) = J_{ohn} \) and \( f_4(PRED) = \ldots \)
Figure 6.4: The linked F-structure fragments derivable via Discard for John swims \(\iff\) Jan zwemt

Figure 6.5 can be generalized still further by the use of the LFG-DOP Discard operator. As we are operating on pairs of f-structures, once Discard has deleted some element from a source f-structure, we need to ensure that all elements of the target f-structure which the deleted element \(\tau\)-supports are also removed in order to prevent non-bona fide linked fragments.

Comparing LFG-DOT2 with DOT Models

With respect to the commit suicide case, we noted in section 5.2.1 that the DOT treebank for the translations in (334), p.179, contained 321 fragment pairs. Without Discard, the equivalent LFG-DOT2 corpus would have the same number of fragments, of course, with associated f-structures added for each c-structure fragment. The three possible LFG-DOT2 derivations for John commits suicide \(\iff\) Jean se suicide are shown in Figure 6.6.

\(John\), i.e. the same element appearing as SUBJ and OBJ in different strings would be treated as if they were different elements.
Figure 6.5: The set of aligned F-structure pairs (no Discard) for *John swims* ↔ *Jan zwemt*
Figure 6.6: LFG-DOT2 derivations for John commits suicide ↔ Jean se suicide
Depending on the LFG-DOP probability model used, the probability of candidate translations differs. Since all of the NPs and verbs in (334) are singular, the competition sets will not change for this feature, but they will for gender. For instance, using competition sets M0 and M1 does not enforce any of the LFG grammaticality checks on-line. In the French treebank, there are 8 determiners and 8 nouns, of which 5 are masculine and 3 feminine. Without reference to the uniqueness condition, any of the nouns can appear with any of the determiners. In a zero-order Markov model, the probability of any noun co-occurring with any determiner is \( \frac{1}{8} \). The same would be true for a PCFG where there was exactly one analysis for a Det N sequence. However, as soon as this differs, then the probability of Det N, \( P(\langle \text{Det}, N \rangle) \), is the sum of the probabilities of all (partial) parse trees generating that sequence. In PCFGs, the probability of an analysis is given in (370):

\[
P(r_1, r_2 \ldots r_n) = \prod_i P(r_i)
\]

That is, the probability of a parse is calculated by multiplying together the probability of each rule \( r \) participating in that parse. However, the probability of word strings is given in (371):

\[
P(w_1, w_2 \ldots w_n) = \sum_T P(T, w_1, w_2 \ldots w_n)
\]

That is, the probability of a string is calculated by summing the probabilities of all the parse trees in which those words appear on the leaves of those trees. These formulae can be contrasted with the probability of a parse tree in STSG given in (27).

Competition sets M2 and M3 enforce LFG’s uniqueness check on-line. Accordingly, the selection of a masculine singular determiner reduces the competition set for nouns from 8 to the 5 masculine singular nouns. Therefore, in trying to produce the wrong, compositional translation, the derivation in (372) will have different probabilities:

![Diagram of parse trees](image-url)
Ignoring the fact that we are producing a translation of *John commits the suicide* (it is the derivation of the wrong target string that we are interested in, after all), under LFG-DOP probability model M1, the probability of this translation is shown in (373):

(373) \[ P(\text{John commits suicide} \leftrightarrow \text{Jean commet le suicide}) = \frac{3}{137} \cdot \frac{1}{8} = \frac{3}{1108} \]

Under LFG-DOP probability model M2, however, the competition set from which a noun is to be selected is constrained by the presence of *le* in the tree. As there are only 5 masculine nouns in the competition set (rather than 8 previously), the probability of the translation now is \( \frac{3}{137} \cdot \frac{1}{5} = \frac{3}{685} \) (or 1.6) times more likely. As a consequence of enforcing the uniqueness check, there are fewer permissible sentences than previously, so all grammatical strings have a higher probability than before. On closer inspection, however, derivation (372) and the subsequent probability calculations are not possible given the treebank derived from the sentences in (334), p.179, as there is no link between the two *suicide* nouns. Rather, the actual DOT link is that shown in (335).\(^{11}\) A slight alteration of S1 in (334) to (374) will now enable this derivation, as *son* will be linked to *his*, thereby allowing a link between the two nouns:

(374) Son suicide est tragique \( \leftrightarrow \) His suicide is tragic

Creating a new treebank results in 336 linked fragments (cf. Table 6.5, p.245, section 6.3.6 below), with an extra 10 s-links, 3 np-links, 1 det-link and 1 n-link. The derivations for the *se suicide* translations will not change as a result, but many more derivations are now possible for the *commet le suicide* alternative. Of course, all the probabilities will change given the new treebank. Retesting the translations in (339), p.181, now gives the new results for the specific translations in (375):

(375) a. \( P(\text{John commits suicide} \leftrightarrow \text{Jean se suicide}) = 0.000637 \approx \frac{1}{1579} \)

b. \( P(\text{Mary commits suicide} \leftrightarrow \text{Marie se suicide}) = 0.0059 \approx \frac{1}{169} \)

As expected, both probabilities decrease slightly compared to (340). For the translation in (339c) there are now 40 derivations with total probability 0.00442 \( \approx \frac{1}{229} \), and for the translation in (339d) there are now 38 derivations with total probability 0.001789 \( \approx \frac{1}{589} \).\(^{12}\) Ranking the translations against each other gives the results in (376):

\(^{11}\) The LFG-DOT2 linked pair for this translation would have f-structures associated with the linked c-structures, of course, as in (372).

\(^{12}\) Note the large increase—from 7 to 38—in possible derivations by quite a small change in the corpus. Note also that despite
(376) a.  \[ P(\text{John commits suicide} \mid T = \text{Jean se suicide}) = \frac{637}{5057} = 0.126 \]
b.  \[ P(\text{John commits suicide} \mid T = \text{Jean commet le suicide}) = \frac{4420}{5057} = 0.874 \]
c.  \[ P(\text{Mary commits suicide} \mid T = \text{Marie se suicide}) = \frac{5900}{7689} = 0.767 \]
d.  \[ P(\text{Mary commits suicide} \mid T = \text{Marie commet le suicide}) = \frac{1789}{5057} = 0.233 \]

Now we see that for \textit{John commits suicide}, the default, compositional translation has increased to about 7 times more likely, whereas for \textit{Mary commits suicide} the specific translation is preferred only about 3.3 times more than the default translation.\textsuperscript{13} Switching to a different LFG-DOP probability model M2, the results achieved can be estimated as 8/5 greater with respect to the corpus than the probabilities achieved with the M1 probability model, but of course this will not alter the relative weight of the translations.

**Summary**

Using LFG-DOP as the source and target language models overcomes the shortcomings of both Tree-DOP and LFG. LFG is considerably more sophisticated than most published DOP language models. DOP models are tree-based which restricts their applicability as general purpose language models. In contrast, LFG handles (almost all) non-surface phenomena with ease.

Using \( \tau \)-equations ensures that the correct translation will be produced in almost all cases. We have seen that DOT1 cannot ensure this. Using \( \tau \)-equations in isolation, as in LFG-MT, necessitates the ranking of a number of output \( f \)-structures by a human expert. Unlike LFG-DOT1, LFG-DOT2 enables automatic ranking, and pruning (if required) of \( f \)-structures. Furthermore, LFG-DOT2 is more robust than LFG-MT, in that Discard can produce generalized fragments which may be able to deal with input outside the scope of the ‘ungeneralized’ LFG-DOT2 corpus. In these cases, LFG-MT has no option but to offer no translation candidates at all. The fragmentation process in DOP if unconstrained renders such models impractical; with LFG-DOP, especially those including the Discard operator, this problem appears to be considerably worse.

We made suggestions in sections 4.4 and 4.5 as to how the Discard operation may be made more practical. We also observed in section 4.6.1 that better pruning should be possible with the linguistically richer LFG-DOT objects than is the case with DOT structures. Consequently, the perception that the fragmentation process causes treebank size to be a ‘bigger’ problem in LFG-DOP may turn out to be unfounded.

We compared LFG-DOT2 to DOT2 for one example. While the actual probabilities of translation differed in the two models, the relative probabilities of candidate translations with respect to each other did not. Furthermore, we showed that depending on the LFG-DOP probability model selected, the probabilities also

---

\textsuperscript{13}One brute-force way in which we can always ensure that a translation derived from specific set of \( \tau \)-equations is preferred over any wrong alternatives derived compositionally from the default equations is to add a large weight to the LFG-DOT structures derived via the specific \( \tau \)-equations. For example, we could simply add 1 to the probabilities in (375) which would ensure that these specific translations outrank any others. Of course, in such circumstances we are no longer dealing with probabilities but rather with levels of ‘conviction’. All \( \tau \)-equations would need to be listed as either default or specific for such an approach to be maintainable. We do not consider such a model further here.
differ. The richer the probability model (in terms of the constraints exercised), the fewer possible strings are permitted, so the probabilities of the sentences which are derivable increase. As before, the relative probabilities of the strings (and translations) do not change.

We demonstrated in chapter 5 that the DOT2 γ function copes with translational phenomena which cause problems for the LFG τ-equations. LFG-DOT2 maintains the τ translation relation to increase the chances of the correct translation being produced. We hypothesized a number of ways in which the two functions might best co-operate, with the γ relation taking priority. Nevertheless, one has to ask the question as to whether it is fruitful to try and integrate the two translation relations. We need the presence of the f-structure information in order to allow Discard to run and thereby make LFG-DOT more robust than LFG-MT. However, Discard can operate whether the f-structures are linked via τ or not, so it would appear that the τ operation itself is not needed. In some cases, as we suggested above, the τ-equations would provide extra evidence in favour of the correct translations, where owing to the restricted nature of DOT-links, only semi-compositional linked fragments are available. Indeed, this flaw influences the changes made in our final translation model, LFG-DOT4.

In the next two models the τ operation is omitted, with the translation relation stated solely in terms of γ.14 As well as allowing further robustness via Discard, we shall see that the functional information contained in the f-structures is also able to constrain the DOT-links sufficiently to prevent the derivation of examples such as (352), p.188. Having proposed the omission of the τ function, we discuss in the next section whether it is nevertheless inferable from other constraints.

6.2.3 LFG-DOT Model 3: Translation via γ with Monolingual Filtering

The LFG-DOT3 model proposed in this section contains the DOT2 links between source and target c-structures, but with additional syntactic functional constraints which prevent ungrammatical structures such as (352), p.188, from being formed, thereby enabling correct translations to be output. The f-structure information can be seen, therefore, as useful for monolingual disambiguation in both source and target sides. Ill-formed or unknown input is still processable by running Discard over the set of linked source and target \( \langle c, \text{LFG-DOP} \phi, f \rangle \) fragments. The LFG-DOT3 architecture is shown in (377):

\begin{equation}
\begin{array}{c}
\text{LFG-DOP-} \phi \\
\downarrow \gamma \\
c \\
\downarrow \\
c' \\
\text{LFG-DOP-} \phi'
\end{array} \quad \begin{array}{c}
f \\
\rightarrow \\
f'
\end{array}
\end{equation}

As an example, let us consider the \textit{just} \( \leftrightarrow \text{venir de} \) headswitching case. In terms of LFG-DOT3, the translation relation is shown in (378):

\begin{equation}
\begin{array}{c}
\text{LFG-DOP-} \phi \\
\downarrow \\
c \\
\downarrow \\
c' \\
\text{LFG-DOP-} \phi'
\end{array} \quad \begin{array}{c}
f \\
\rightarrow \\
f'
\end{array}
\end{equation}

14 We leave for future work the question as to whether this approach is fruitful for languages which differ significantly at the level of surface structure, e.g. English and Warlpiri. In such cases, perhaps an LFG-DOT1 or LFG-DOT2 model may be better to relate translational equivalents at the level of f-structure rather than c-structure.
The γ link between semantically equivalent elements in the source and target c-structures can be seen on the VP nodes. As in DOT2, *fell* is not considered to be semantically equivalent to *tomber* owing to their different FIN(ite) values, added to the fact that *fell* has a TENSE value whilst *tomber* does not. Hence this translation fragment can only be reused by substituting this pair with associated singular NP subjects at the appropriate nodes in an S-linked fragment. In this respect, as with DOT2 (and LFG-DOT2), this LFG-DOT3 model continues to suffer from limited compositionality. We address this concern further in the next section which deals with the LFG-DOT4 model, which has an extra level of processing called ‘Extended Transfer’. In all other respects, LFG-DOT3 and LFG-DOT4 are the same models, so while we end up rejecting LFG-DOT3 in favour of LFG-DOT4, much of the ensuing discussion is relevant to our final choice of model, LFG-DOT4.

**Can LFG-DOT3 Improve on LFG-MT and CAT?**

Having just shown how LFG-DOT3 copes with a headswitching problem, we shall now investigate whether LFG-DOT3 can handle translation examples which the LFG-MT and CAT models find problematic. Let us use the examples in (379):

(379) a. I think that the baby just fell ⇔ Je pense que le bébé vient de tomber.

    b. Le gouvernement sait permettre que les étudiants le fassent ⇔ The government knows how to permit the students to do it.

The first here (a repetition of (89)) is an example of embedded headswitching which causes problems for LFG-MT, as the default τ-equations in the entry for *think* (cf. (92), 76) clash with those on the structural rule for ADVP (cf. (87)), as we showed in (94)–(98). The second is a case where two exceptional translational phenomena are combined in the one translation case, which is problematic for CAT.
Figure 6.7: LFG-DOT3 representation for *I think that the baby just fell* ↔ *Je pense que le bébé vient de tomber*.

The LFG-DOT3 representations for the full trees in (379a) are shown in Figure 6.7. Source and target trees are linked at the topmost S, NP, V and VP levels, as well as at Š, COMP and embedded S levels. Given that each source fragment will be linked to its target counterpart with the same label, each (*source*, *target*) linked pair can be deleted to make the new linked fragment pair in (380):
The trees in (380) are linked at S, NP, VP, DET and N levels. Once these are deleted, the remaining fragments are linked at embedded VP level, exactly as in (378). That is, embedded headswitching cases in LFG-DOT3 are dealt with in exactly the same manner as non-embedded headswitching cases. Given that fragments such as (378) and (380) exist, such complex cases can also be dealt with compositionally, as (381) illustrates:
That is, the lower linked source and target sentence pair is substituted into the source and target S-nodes in the upper trees. At the same time, their f-structures are merged with the COMP f-structures of the source and target f-structures respectively.

Turning now to (379b), we see that this example combines two problematic translational phenomena: the translation of savoir into know how (i.e., the adverb how has to be ‘shoehorned in’), as well as the fact that an argument (étudiants) of a lower verb (faire) in French is raised to become the object (students) of a higher verb (permit) in English. Way et al. (1997:354) observe that the savoir rule encroaches on the permettre rule in CAT which necessitates the writing a new rule for this combination of exceptions only. The source and target top-level objects in LFG-DOT3 for (379b) (with a simplified treatment of pronominals) can be seen in Figure 6.8.

As well as at the topmost S-level, links can be made between subject NPs, and their content. This reduces the statement of the minimal translation relation to inside the outermost VPs. What links are possible inside these elements? The topmost Œ’s can be linked, and within these constituents, we can link ⟨permettre, permit⟩, ⟨les étudiants, the students⟩, and ⟨le fassent, do it⟩. The minimal statement of the translation relation for (379b) in LFG-DOT3, therefore, is (382):
Figure 6.8: Simplified LFG-DOT3 representation for *Le gouvernement sait permettre que les étudiants le fassent* $\leftrightarrow$ *The government knows how to permit the students to do it*
The CAT rule built to handle this combination of exceptional phenomena needs to stipulate the interaction between savoir → know how and permettre que + S → permit NP + V, in the same rule. One can see that this is not necessary in LFG-DOT3: there is no mention of permettre or permit at all in (382). Note also how the remaining f-structure content will constrain the ability of other fragments to combine with this statement of the translation relation. The NUM values on the NPs rather restrict possible combinations, but we assume that larger corpora will enable these restrictions to be overridden. Turning now to the verbs, only French verbs which take a COMP and which map onto English verbs which subcategorize for an OBJ and an XCOMP are capable of being inserted at the relevant nodes in the source and target trees. Given this, (382) would combine with, for instance, (vouloir, want), as in (383):

(383) Le garçon sait vouloir que les filles l’aiment ← The boy knows how to want the girls to like him.

LFG-DOT3 can, therefore, cope with certain cases of combinations of exceptional phenomena which prove problematic for CAT. It can also be seen quite clearly from examples such as (382) that LFG-DOT3 has no ‘local tree restriction’: there is no necessity to mention the subject NP nodes at all, despite their being sisters to the VP. In CAT, this subject slot needs to be included, notwithstanding the fact that it is peripheral to the translation relation (cf. (18), p.30).

Nevertheless, it appears that some of the problematic cases for CAT pose similar difficulties to LFG-DOT3. For instance, one complex case with which LFG-DOT3 seems to have similar problems to CAT is (384):

(384) On sait le faire ← It is known how to do it.

This example (seen previously as (17c), p.29) illustrates the combination of two complex transfer cases: the savoir ← know how problem seen above, as well as the fact that French sentences beginning with on are often best translated as passives in English. The top-level LFG-DOT3 representation of this complex case is shown in Figure 6.9, with simplified f-structure treatments of pronouns.
Figure 6.9: Simplified LFG-DOT3 representation for *On sait le faire* ↔ *It is known how to do it*

These fragments are obviously linked at the sentential level. Disregarding fragments produced by *Discard*, the only other *(source, target)* links which can be made are those at the level of the lowest VP, and within these VPs. That is, we can link *(le, it)* and *(faire, do)*. The minimal statement of the translation relation in LFG-DOT3 for (384) is, therefore, (385):

It must be admitted that (385) is very close to a notational variant of the *CAT* rule (cf. Way *et al.*, 1997:353) for (384). *CAT*'s problem with this particular translation is its local tree restriction. It appears that LFG-DOT3 faces similar problems, for (384) at least: given that no other γ links are possible, (385) is the minimal statement of the translation relation for this example in LFG-DOT3. In defence of the LFG-DOT3 model, it is clear that this translation example is one where wholesale changes are made, so it is perhaps unsurprising that the translation relation cannot be broken down any further than (385).
In sum, therefore, it appears that LFG-DOT3 can solve some of the translational phenomena which are problematic for LFG-MT. It can also solve some of the combinations of exceptional phenomena which cause problems for the CAT model. In other instances of similar problematic phenomena, however, its solutions may be considered less than optimal.

**Can the $\tau$ function be reconstructed?**

It can be seen in both (377), p.210, the schematic representation of Model 3, as well as the particular translation examples (378)–(385), that the translation relation is stated on $c$-structures. Unlike LFG-DOT2, there are no $\tau$-linked $f$-structure fragments, as the $\tau$-mapping has gone. Recall that in section 3.1, we described how Johnson (1988) made explicit the important distinction between constraints and mere graphical representations of the same. In what follows, we discuss whether certain constraints ($\tau$-equations) can be inferred from others ($\phi$ and DOT links). This is indeed the case for simple transfer cases, such as the love $\leftrightarrow$ aimer example in (386):

![Diagram](image)

From (386), it can be seen that the $\phi$ and DOT links are those in (387):

(387) English phi links:
phi(S1,f1), phi(NP1,f2), phi(VP1,f3), phi(V1,f3), phi(NP2,f4)

French phi links:
phi(S2,f5), phi(NP3,f6), phi(VP2,f7), phi(V2,f7), phi(NP4,f8)

DOT links:
dot(S1,S2), dot(NP1,NP3), dot(VP1,VP2), dot(V1,V2), dot(NP2,NP4)

That is, the sentential node S1 in the English tree in (386) maps via $\phi$ onto the outermost bracket $f_1$ in the corresponding English $f$-structure. We assume an $\uparrow=\downarrow$ functional annotation on $c$-structure verbal nodes.
Consequently such c-structure nodes map onto the same f-structure values as their mother nodes.

Now, if we know that $\phi_i(NP_1, f_2)$ and $\phi_i(NP_3, f_6)$, and also that $\text{dot}(NP_1, NP_2)$, then we can deduce the $\tau$-equation $\tau(f_2, f_6)$. The same can be done for all other $\phi$- and DOT-linked elements in (386), in which case we end up with the set of $\tau$-equations for the $\text{love} \leftrightarrow \text{aimer}$ simple transfer case in (388):

$$\tau(f_1, f_5), \tau(f_2, f_6), \tau(f_3, f_7), \tau(f_4, f_8)$$

This is exactly the set of $\tau$-equations one would obtain if the $\tau$-mapping was stated explicitly. It is clear that such inference extends to all cases of simple transfer. If this extends to other more complex translation cases, we could axiomatize the $\tau$ function and end up with a description of the translation relation as in (389):

$$\tau\text{-axiom:}$$

$$\forall C_s, C_t, F_s, F_t([\phi_i(C_s, F_s) \& \phi_i(C_t, F_t) \& \text{dot}(C_s, C_t)] \Rightarrow \tau(F_s, F_t))$$

Here $C_s$ refers to the c-structure of the source language, $C_t$ refers to the target c-structure, $F_s$ refers to the f-structure of the source language, and $F_t$ to the target f-structure. If we now add the $\tau$-axiom in (389) to our $\phi$- and dot-links for a particular translation pair, then an explicit linguistic description $LD$ is simply the union of all phi- and dot- and tau-atoms that can be inferred from $\Phi$, $\text{Dot}$ and the $\tau$-axiom, as in (390):

$$\Phi \cup \text{Dot} \cup \tau\text{-axiom} \vdash LD$$

Here $\Phi$ is the set of phi-links and $\text{Dot}$ is the set of dot-links (produced via $\gamma$) for a pair of source and target $(c, \text{LFG-DOP-$\phi$}, f)$ structures, as in (387), for instance. The $\tau$-axiom is that given in (389).

Let us now examine whether the $\tau$-axiom holds with more difficult instances of translation such as relation-changing cases. We can illustrate this with the $\text{like} \leftrightarrow \text{plaire}$ example shown in (391):
The $\phi$- and DOT-links obtainable from (391) are those in (392):

(392) English phi links:

$\phi(S1, f_1), \phi(NP1, f_2), \phi(VP1, f_3), \phi(V1, f_3), \phi(NP2, f_4)$

French phi links:

$\phi(S2, f_5), \phi(NP3, f_6), \phi(VP2, f_7), \phi(V2, f_7), \phi(P1, f_8), \phi(NP4, f_9)$

DOT links:

$\text{dot}(S1,S2) \text{ dot}(NP1, NP4) \text{ dot}(NP2, NP3)$

Similarly, we know that $\phi(NP1, f_2)$ and $\phi(NP4, f_9)$, and also that $\text{dot}(NP1, NP4)$, so the $\tau$-equation $\tau(f_2, f_9)$ is inferable. Continuing the process for all other $\phi$- and DOT-linked elements in (391), the set of $\tau$-equations for the $\text{like} \leftrightarrow \text{plaire}$ relation changing case in (393) is obtained:

(393) $\tau(f_2, f_9), \tau(f_1, f_3), \tau(f_4, f_6)$

The first of these shows that $\text{Mary}$ translates as $\text{Marie}$, the second indicates that the two sentences are translations of each other, and the third equation shows the translation of $\text{John}$ to be $\text{Jean}$. We do not, however, have an explicit equation stating that $\text{likes}$ translates as $\text{plait} \, \text{a}$. The trouble is that the DOT-link stops at a level (or levels) above that required to state the translation relation. Given this, it is unsurprising that the inference of $\tau$-equations does not work for headswitching either, as illustrated by the $\text{like} \leftrightarrow \text{grouag}$ case in (394):
The φ- and DOT-links obtainable from (394) are those in (395):

(395) English phi links:
\[ \text{phi}(S1, f_1), \text{phi}(NP1, f_2), \text{phi}(VP1, f_3), \text{phi}(V1, f_3), \text{phi}(V2, f_4) \]

Dutch phi links:
\[ \text{phi}(S2, f_5), \text{phi}(NP2, f_6), \text{phi}(VP2, f_7), \text{phi}(V2, f_7), \text{phi}(ADV, f_8) \]

DOT links:
\[ \text{dot}(S1, S2), \text{dot}(NP1, NP2), \text{dot}(VP1, VP2) \]

The only \( \tau \)-equations that can be derived from these equations are \( \tau(f_1, f_5), \tau(f_2, f_6), \tau(f_3, f_7) \), i.e. at the sentential, subject NP and VP levels. We cannot obtain any translational information about the content of the two VPs, which is, of course, what constitutes the translation relation itself. It should be clear that the \( \tau \)-axiom inference hypothesis does not work for embedded headswitching for the same reasons. While it works for simple cases, ultimately the inference hypothesis is flawed. Of course, this is indeed what we want to happen: if we were able to infer the complete set of \( \tau \)-equations from φ- and DOT-links, then the LFG-DOT3 model proposed here would equate to the LFG-DOT2 model, which explicitly contained the \( \tau \)-function in its statement of the translation relation. In Model 3, the f-structure constraints are used for monolingual disambiguation only; they do not figure in the statement of the translation relation at all.

Is C-structure the Right Level at which to state the Translation Relation?

A final, remaining question concerns the level at which we are postulating the translation relation: how do we reconcile the fact that surface structure was abandoned as a level at which translation could be described in favour of more abstract models, with our decision to link translational equivalents using the \( \gamma \) function? Of course, both LFG-DOT3 and 4 contain a level at which functional structure is stated, but this plays little role in the translation process per se. Functional constraints are used for monolingual filtering only.
A piece of empirical evidence in favour of so doing is that both models have the potential to get the headswitching cases right, whereas we have shown that LFG-MT cannot cope with these examples using either f-structure or s-structure on which to model the translation relation. Why were trees abandoned in the first place? We know that firstly, any linguistic formalism based purely on PS-trees will be unable to handle certain linguistic phenomena. Secondly, we showed in section 1.3.3 that when faced with combinations of exceptions, the CAT system approximates in the worst case to a sentence dictionary. Part of this criticism can also be levelled at DOT2 (and LFG-DOT models 2-3, but not LFG-DOT4), but at least models 3 and 4 get them right and the ‘rules’ are there in the first place, having been derived in the fragmentation process: in CAT, these rules (one per combination of an exceptional case with any other) had to be written down by a linguist. Given this, it is perhaps not overstating the case to suggest that LFG-DOT models (and DOT2) have ‘learnt’ these hard cases of translation.

Furthermore, it is well known that languages are ‘more similar’ at the level of interlingua, but that does not mean that interlingual solutions to the wide range of translational phenomena dealt with here can be found, nor that interlingual systems abound to any great extent. In any case, almost all systems claiming to be interlingual interpose a level at which bilingual transfer is done in a lookup procedure, akin to transfer-based systems.

From a theoretical viewpoint, the answer is quite clear. CAT fails because of its local tree restriction. LFG-DOT3 (and LFG-DOT4) does not have to relate local trees to local trees: some paired fragments will mirror CAT rules, but others will be quite different objects entirely. It is this fact which contributes to what may be seen as a surprising result, namely that LFG-DOT models of translation outperform tree-based systems such as DOT and CAT, as well as constraint-based systems such as LFG-MT.

**Summary**

LFG-DOT1 expresses the translation relation by means of the $\tau$ mapping, whereas LFG-DOT2 combines $\tau$ with the $\gamma$ function. LFG-DOT3, meanwhile, eschews the $\tau$ relation and relies completely on $\gamma$ to relate translationally equivalent source and target fragments. The $\tau$ relation was abandoned for several reasons:

- it is unable to always express the correct translation relation;
- the $f$-structures produced may not be acceptable to the target grammar;
- the $\gamma$ relation is capable of producing correct translations where this cannot be guaranteed with $\tau$.

Despite the deliberate omission of the $\tau$-equations, we investigated whether they were nevertheless retrievable via other constraints. We demonstrated that this is not so, indicating clearly that LFG-DOT3 is different in nature from LFG-DOT2.

We also showed that LFG-DOT3 is able to cope with some of the translation examples which cause problems to the CAT translation model. Despite stating the translation relation between trees, LFG-DOT3 does not suffer from the local tree restriction which causes CAT to approximate to a sentence dictionary when dealing with combinations of hard cases.
The presence of functional information in LFG-DOT3 prevents ill-formed structures such as (352), p.188, from being formed, unless Discard is used to process such linked pairs in dealing with ill-formed input. LFG-DOT3, therefore, has a notion of grammaticality which is missing from DOT2. Importantly, this can be used to guide the probability models in the manner required.

In using γ to express the translation relation, we observed that LFG-DOT3 is open to the same criticism as DOT2 in that both models suffer from limited compositionality. Given this, the minimal translation relations cannot be stated. Therefore an LFG-DOT3 database will need to be extremely large in order for such fragments to have even a small probability of participating in the combinatory process with other fragments. LFG-DOT4 does not suffer from this drawback, rendering the likelihood of its fragments’ usefulness as translationally relevant examples significantly higher.

6.2.4 LFG-DOT Model 4: Translation via γ and ‘Extended Transfer’

In the previous section, we observed that the outstanding problem with Model 3 is its retention of the DOT2 problem of limited compositionality. Returning to the just ↔ venir de headswitching case (378), p.210, we would like to be able to ‘relax’ some of the constraints in order to map ⟨fell, tomber⟩ to make these linked fragments more general, and hence more useful. In so doing, we would remove this problem of limited compositionality.

In LFG-DOT4, the basic translation relation is expressed by γ, as with LFG-DOT3. In LFG-DOT4, however, there is a second application of Discard, by which ‘lemmatized’ forms are arrived at on which ‘extended transfer’ can be performed. Discard relaxes constraints in order to produce a set of generalized fragments with the potential to deal with ill-formed or unknown input. Once the TENSE and FIN features have been relaxed on the lowest verbs in both fragments in (378), they can be regarded as translationally equivalent. Given this, ⟨fell, tomber⟩ are linked and lemmatized, as in (396):

![Diagram](image)

Now that ⟨FALL, TOMBER⟩ are linked, they can be deleted to produce the generalized form of the translation relation, namely (397):

```
```
If fragment pairs such as (397) prove subsequently to be of use in combining with other fragments, any resultant translation will be marked as ungrammatical with respect to the corpus, given that Discard was used in its derivation. Nevertheless, even if we restrict the impact of Discard on the probability space (cf. section 6.3.6. below), such translations will receive some probability, whereas the semi-compositional variants from which they were derived may not be able to produce any translation.

The examples in (396) and (397) are, however, a little incomplete. The reader will observe that we have been rather selective in our choice of lemmatization. Given that the lemmatized forms in these examples are peripheral to the translation relation, we have chosen simply to apply Discard to these elements so that they may be deleted. In practice, if all verbs in (378), p.210, are subjected to the lemmatization process, we will instead end up with a different linguistic object, namely (398):

This remains unproblematic given that there is no γ-link between VENIR and a node in the English tree. Therefore, we are still able to link (TOMBER,FALL) and end up with the desired, generalized translation
relation \( \text{just, VENIR de} \).\(^{15}\)

*Discard* is of course not restricted to acting upon verbal features. Do we require the NUM features to be removed also, or should they be protected? Note that if all verbs are lemmatized, as in (398), then there is no harm in removing features on the NPs to generalize the fragment further. However, having examined further ‘hard’ translation cases, especially those containing combinations of exceptions, we feel that a restriction of *Discard* to verbal features in the extended transfer phase may be required. Let us investigate the translation *On le fait* \( \leftarrow \) *It is done* example in (399):

Obviously, one required link between lemmatized forms is \( \text{\textsc{faire}, DO} \), as shown. Once such a link is made, and these forms are deleted, ‘sentences’ such as *On le VOIR* can be ‘translated’ as *It is SEE*. It is not difficult to see that this output can be readily post-edited into the required translation *On le voit* \( \leftrightarrow \) *It is seen*. However, if *Discard* is allowed to run in an unconstrained manner, then the NP features will be able to be deleted.

Prior to exemplifying a potential problem with this, the features in the NP f-structures require some dist-

\(^{15}\)We retain lemmatized forms in the ‘translations’ produced for post-editing. Any other format more suitable to effective post-editing may, of course, be chosen. We consider below whether the target grammar might be able to complete such partial f-structures and thereby avoid the need for post-editing.
cussion. It is an matter of ongoing debate in the LFG community as to what information should occur in f-structure. With respect to the pronouns in (399), we have chosen to include a grammatical gender marker in the f-structure of the French OBJ pronoun, while those associated with English pronouns are instead considered to be referential gender markers. On this view, *le* is encoded as GEN=MASC. *On*, however, is regarded as being unmarked for grammatical gender and is shown as it is used impersonally (that is, it does not bind to anything in the discourse, syntactically). On the English side, *it* is not used with any grammatical function but rather is used referentially. Depending on the context, *it* may be used impersonally, of course.

Ignoring other partial fragments which are peripheral to the argument here, one possible pair of linked f-structure fragments will be those in (400):

\[
\begin{align*}
(400) & \quad \text{NP} \quad \text{CASE} \quad \text{NOM} \quad \text{NP} \\
& \quad \text{on} \quad \text{PRED} \quad \text{`pro'} \\
& \quad \text{NUM} \quad \text{SG} \\
& \quad \text{PERS} \quad 3 \\
& \quad \text{GEN} \quad \text{unmarked} \\
& \quad \text{NP} \quad \text{CASE} \quad \text{NOM} \\
& \quad \text{it} \\
& \quad \text{PRED} \quad \text{`pro'} \\
& \quad \text{NUM} \quad \text{SG} \\
& \quad \text{PERS} \quad 3 \\
& \quad \text{GEN} \quad \text{unmarked}
\end{align*}
\]

This is brought about by deletion of the REF features which render the two subject NPs as translationally equivalent. Given this, we can draw a link between the \(\langle \text{on}, \text{it} \rangle\) nodes in (399). The (FAIRE, D0) lematized nodes are already linked, so now both pairs of linked nodes can be deleted, leaving the remaining \(\gamma\)-linked pairs in (399). The (FAIRE, D0) lematized nodes are already linked, so now both pairs of linked nodes can be deleted, leaving the remaining \(\gamma\)-linked pairs in (401):

\[
\begin{align*}
(401) & \quad \text{NP} \quad \text{VP} \quad \text{NP} \quad \text{VP} \\
& \quad \text{le} \quad \text{is} \quad \text{V} \\
& \quad \text{V} \quad \text{V} \\
& \quad \text{NP} \quad \text{VP}
\end{align*}
\]

Ignoring the details of the resultant f-structures, it is obvious that such a link ought not to be allowed to exist. This came about by permitting Discard to remove NP features in the extended transfer phase. Given the need to avoid structures such as (401), we propose to allow Discard to delete only verbal features in the extended transfer phase. One might like to make a case for \(\langle \text{le}, \text{it} \rangle\) in (399) to be linked, in which case on \text{les} \text{VOIR} would be ‘translatable’ as They are SEE, assuming a prior linking of \(\langle \text{les}, \text{they} \rangle\). However, such a link would not be made in the ‘regular’ LFG-DOT treebank, as these items would not be considered to be translational equivalents—one is subject, the other object. The only place this might happen would be in extended transfer. Nevertheless, because of possible linked elements such as (401), we do not permit these NPs to be linked (by relaxing the CASE and GEN features).\(^\text{16}\) While the set of features which may be

\(^\text{16}\text{Following extended testing with suitable corpora, it may turn out that such undesirable linked elements as (401) play such a minimal role in the overall calculation of the probabilities of translation candidates that we might allow Discard full rein in the extended transfer phase. This remains an open question at this juncture.}\)
relaxable by *Discard* probably needs to be determined empirically, there are some features which definitely ought to be excluded. Sadler (personal communication) points out that this set ought to include at least any attribute whose value is (+/-). Removing such features from structures will obviously cause their polarity to be altered, which will most likely result in mistranslations being produced.

In the same way, this limits to a small degree the combination of exceptions case in (402):

(402) On vient de le faire $\leftrightarrow$ It has just been done.

Once we relax the TENSE features of the verbs in (402), the $(c, \text{LFG-DOP-} \phi, f)$-linked pairs in (403) will be derived:

Such a fragment pair would be able to ‘translate’ On *VENIR de le VOIR* $\leftrightarrow$ *It HAVE just BE SEE*, for example. This can readily be post-edited into the required translation *It has just been seen*. Similarly, (385) will be generalizable once TENSE and PASSIVE constraints are relaxed, enabling *(sait,known)* to be linked and deleted.

Rather than relying on a post-editing phase to transform cases such as On *VENIR de le VOIR* $\leftrightarrow$ *It HAVE just BE SEE*, one wonders whether the target grammar might ‘fill in’ any missing features and thereby enable a
correct, target string to be formed. This is what target grammars have to do in any case, namely coerce
target structures in generation, cf. (15), p.28, for the Eurotra EF translation system, and (76)–(80), p.71f.,
for LFG-MT. The quote of Kaplan et al. (1989:276; cf. p.71 above) is apposite, in that the objects input to
the generation phase necessarily have to be augmented by rules in the target grammar so as to produce a
bona fide target representation from which a well-formed target string can be read off.

What we want to do is allow the target grammar to fill in features (but not overwrite them) only in f-structures
produced by Discard. With respect to the target structures in (399), for instance, the only element which
needs to be fleshed out is the D0 node in the c-structure. In the linked f-structure, the PRED information
is available, in addition to the TENSE and PASSIVE values. In a treebank of any size, we can hypothesize
that there will be many other f-structures which are subsumed by an f-structure with these minimal features
present, all of which will be linked to their respective target c-structures, some of which we can expect to
have the correct surface realization of these features, namely done. If this lexeme occurs with a reasonable
frequency (to be arrived at empirically), then a final processing stage can be foreseen whereby the word form
done replaces the D0 form in the output target c-structure. This may not be possible in all cases (cf. (403)
below), but it will be in some, at least.

In bringing this section to a close, we feel it is important to convince the reader that mistranslations cannot
occur. For instance, one might imagine that once TENSE features are deleted, then it may be possible to
translate John swims \(\leftrightarrow\) Jean a nagé. However, with the envisaged lemmatization process, this can only
occur if the post-editor chooses an inappropriate morphological form to represent any particular tense value.
In this simple example, let us assume that John swims is correctly linked to Jean nage. After Discard has
deleted TENSE features, we would end up with the linked forms \(\langle\text{SWIM, NAGER}\rangle\). The pair \(\langle\text{John, Jean}\rangle\)
are dealt with in the normal way. Note then, that the lemmatization process envisaged here differs significantly
from the approach taken in the Candide system (Brown et al., 1992), which reduces morphological ‘cousins’
to a shared lemma format in order to gather better statistics for their language models. We restrict lemmat-
ization to the extended transfer phase; we do not envisage a reduction to lemmas from the outset. Instead,
we advocate a multi-layered approach, where ‘regular’ LFG-DOP techniques are applied first. Then in a
second level, using Discard, we arrive at lemmatized forms which extend the applicability of linked fragments
to other translations. Any translations obtained are viewed as ‘ungrammatical with respect to the corpus’
(whether they are correct or not) as they were obtained using Discard. Nevertheless, the lemmatization
process overcomes the limitation of DOT2 and previous LFG-DOT models, namely limited compositionality.

Summary

Most translation examples such as (378)–(385) will be handled in LFG-DOT4 in exactly the same way as in
LFG-DOT3. The outstanding problem with LFG-DOT3, however, is its retention of the DOT2 problem of
limited compositionality. LFG-DOT4, like LFG-DOT3, expresses the basic translation relation by means of the
\(\gamma\) function. LFG-DOT4 adds an ‘Extended Transfer’ step to LFG-DOT3 by producing lemmatized forms
using a second application of Discard. This extension overcomes the problem of limited compositionality.
Lemmatisation (via Discard) barely affects the probability of previous, correct LFG-DOT translations which
were arrived at in a limited compositionality manner. We propose again the use of Good-Turing to limit

228
the scope of any newly created translations via lemmatization by restricting the amount of probability space available to the extended transfer phase. How much of this probability space should be given over to the lemmatized translation pairs needs to be established empirically via extensive testing when appropriate corpora become available. Despite the fact that lemmatized translations will obtain only small probabilities, the very fact that they are obtained and offered up as translation candidates is what is important: for some of the more complex cases, almost the exact translation example needed to be found in LFG-DOT3 for a translation candidate to be obtained. Extended transfer generalizes translation fragments considerably, thereby increasing the likelihood that such fragments will indeed play a role in the derivation of new translations.

Finally, we hypothesized a further stage of processing in which grammatical target f-structures might be formed from those generated via Discard in the extended transfer phase of translation, by reference to other similar f-structures in the treebank. In this way, correct morphological forms may replace lemmatized forms so that correct target strings are generated, thereby obviating the need for post-editing of such output.

6.3 Probability Models for LFG-DOT

We present here probability models for each of the LFG-DOT architectures described above. We also summarize probability models which have been given for DOP-based models, other than the basic relative frequency estimator of Bod (1992; 1995; 1998).

6.3.1 Model 1: Translation via $\tau$

LFG-DOT1 uses LFG-DOP to describe its source and target language models. The translation process relates (source, target) f-structures via the LFG translation relation $\tau$. We showed that while LFG-DOT1 improved over both LFG-MT and DOT1 models of translation, it maintained many of the inherent faults of LFG-MT. Nevertheless, given that it remains a model of translation, the probability model on which it is based needs to be described.

Given a source string $W_s$, we need to find the target string $W_t$ which is its most likely translation. Chang & Su (1993) provide the probability model in (404) for a speech-speech translation system:

$$P(W_t \mid A_s) = \sum_{W_s, C_{st}} P(W_s \mid A_s).P(C_s \mid W_s, A_s).P(C_t \mid C_s, W_s, A_s).P(W_t \mid C_t, C_s, W_s, A_s)$$

$A_s$ is the source speech signal, $W_s$ is the source string, $W_t$ the target string, $C_s$ the representation of the source language and $C_t$ the target representation. Alshawi (1996) notes that statistical translation models like (404) will more than likely suffer from the sparse data problem. That is, there will (in all probability) never be enough data to model these numbers with any reasonable accuracy owing to the small size of the data set. Alshawi applies, therefore, a number of first order Markov assumptions to (404) in order to approximate these probabilities, as in (405):

$$P(W_t \mid A_s) \approx \sum_{W_s, C_{st}} P(W_s \mid A_s).P(C_s \mid W_s).P(C_t \mid C_s).P(W_t \mid C_t)$$

Comparing (405) with (404), the assumptions made are as follows:
that the probability of the source representation $C_s$ is conditional only upon the probability of the source string $W_s$, and not the prior probability of the joint events $W_s$, $A_s$;

that the probability of the target representation $C_t$ is dependent solely on the prior probability of the source representation $C_s$, and not on the prior probability of the joint events $C_s$, $W_s$, $A_s$;

and that the probability of the target string $W_t$ is dependent solely on the prior probability of the target representation $C_t$, and not on the prior probability of the joint events $C_t$, $C_s$, $W_s$, $A_s$.

All of these assumptions are, of course, wrong: with regard to the latter assumption, for instance, the probability of the target string is influenced by the probabilities of the source signal, the string associated with it, and its representation. Nevertheless, the calculations involved become more manageable and the statistical models more reliable, perhaps somewhat surprisingly. This is because the frequency of events increases when fewer random variables are considered, so the statistical modelling process becomes more robust as there is a higher ‘confidence factor’ in the numbers involved. For example, a bigram model often outperforms a related trigram model as the bigram model factors in a far larger number of events than the corresponding trigram model. Of course, there are a number of smoothing methods such as linear interpolation and backoff models which help overcome problems with data sparseness.

Ashlavi (op cit.) then needs to use Bayes’ theorem to manipulate the probability model in (405) for a word string $W_s$ given an acoustic signal $A_s$, $P(W_s \mid A_s)$, in order to produce instead a model for $P(A_s \mid W_s)$. As the input into his speech translation system is an acoustic signal, it is the task of the next component (a speech recognizer) to decode the signal into a string. The prior probability, therefore, is the acoustic signal $A_s$, but it is impossible to estimate a string given such a signal, hence the need to manipulate the probability model.

However, in our LFG-DOT models we need not concern ourselves with acoustic signals. Hence the probability model for a speech–speech translation system in (404) can be amended to the model suitable for a text-based translation system in (406):

\[
(406) \quad P(W_t \mid W_s) = \sum_{R_s} P(R_s \mid W_s) \cdot P(R_t \mid R_s, W_s) \cdot P(W_t \mid R_t, R_s, W_s)
\]

That is, the conditional probability of the target string $W_t$ given the source string $W_s$, namely $P(W_t \mid W_s)$, is equal to the probability of the representation $R_s$ of the source string given that string, multiplied by the probability of the target representation $R_t$ given the prior probabilities of the source representation and string, multiplied by the probability of the target string $W_t$ given the prior probabilities of the source and target representations and the source string.

For similar reasons to those pointed out above, the probability model in (406) needs to be adapted to try to avoid the problem of sparse data—in our case, LFG-DOT fragments. We assume, therefore, a first order Markov assumption to approximate these probabilities, as in (407):

\[
(407) \quad P(W_t \mid W_s) \approx \sum_{R_s} P(R_s \mid W_s) \cdot P(R_t \mid R_s) \cdot P(W_t \mid R_t)
\]

That is, comparing (407) with (406), the assumption made is that the probability of the target string $W_t$ is dependent solely on the prior probability of the target representation $R_t$, and not on the prior probability of
the joint events $R_t$, $R_s$, $W_s$. The other simplification, however, does not incorporate Markov assumptions—it can be made given the nature of the linguistic representations we are using in our LFG-DOT models. Note that a full LFG-DOP ($e$, LFG-DOP-$\phi$, $f$) representation $R_s$ of course includes the string $W_s$, as this occurs in the c-structure itself. In this case, therefore, $P(R_t \mid R_s, W_s)$ is always equal to $P(R_t \mid R_s)$, a second simplification we have made in amending (406) to (407). While this simplification is valid, the first approximation is of course wrong: the probability of the target string is influenced by the probability of the source string, but we choose to omit this information so as to simplify and simultaneously improve our translation model.

Returning to (407), for similar reasons as just outlined, one might wonder whether $P(W_t \mid R_t)$ may be omissable since it is trivially equal to 1, since $W_t$ is always included in $R_t$. We shall make use of this observation in our probability models for LFG-DOT2, but we cannot do so for LFG-DOT1. The output from the transfer phase is a target f-structure, from which many possible strings may be generated (cf. free word order languages, for instance). In addition to outputting a target f-structure, transfer in LFG-DOT2 also includes the production of the target c-structure tree via $\gamma$, a constituent part of which is obviously the target string itself. In LFG-DOT2, therefore, we omit the $P(W_t \mid R_t)$ term (cf. (409) and following models), but this must be maintained in our probability models for LFG-DOT1.

Determining the $W_t$ that maximizes $P(W_t \mid W_s)$ involves making the following calculations if the probability model in (407) is used:

- $P(R_s \mid W_s)$: the source language LFG-DOP model, which can be calculated by the LFG-DOP model (Bod & Kaplan, 1998), and then by normalizing. That is, $P(R_s \mid W_s) = \frac{P(R_s, W_s)}{\sum_{R'_s} P(R'_s, W_s)}$.

- $P(R_t \mid R_s)$: transfer, in terms of $\tau$-equations, which can be estimated by $\frac{P(R_t, R_s)}{P(R_t)}$, i.e. dividing the probability of the f-structures $R_t$ and $R_s$ together, by the probability of $R_s$.

- $P(W_t \mid R_t)$: the target language LFG-DOP generation model, where $P(W_t \mid R_t) = \sum_{W_t} \frac{P(R_t, W_t)}{P(R_t)}$.

This is the main LFG-DOT Model 1 presented in section 6.1.1. Recall, however, that the second version of LFG-DOT1 in (354), p.192, terminates with the output of a target f-structure; the process of generating a target string is assumed to be outside the scope of this translation model. Given this, assuming separate language corpora, we want to find the most likely target f-structure $f'$ given a source c-structure $c$ and f-structure $f$, the mapping between them LFG-DOP-$\phi$, and the tau-equations $\tau$. Given the need for these components, we can formulate the probability of the target f-structure $R_t$ being the translation of the source string $W_s$ as (408):

$$
\begin{align*}
\text{Max}_{R_t} P(R_t \mid W_s) &= \text{Max}_{R_t} \sum_{R_s, W_s} P(R_s \mid W_s) \cdot P(R_t \mid R_s, W_s) \\
&= \text{Max}_{R_t} \sum_{R_s} P(R_s \mid W_s) \cdot P(R_t \mid R_s)
\end{align*}
$$

(408)

As with (407), we are able to simplify the $P(R_t \mid R_s, W_s)$ term to $P(R_t \mid R_s)$, as the source language

\footnote{Note that a target language LFG-DOP model is able to compute the most likely string among all possible strings compatible with any such f-structure.}
LFG-DOP model includes the source string $W_s$ in the $c$-structure tree.\footnote{Note that if we were to posit a translation model merely between source and target f-structures, with no recourse to $c$-structure information nor the LFG-DOP-$\phi$ function, then we would need to incorporate a Markov assumption that the target f-structure’s derivation is independent of the original words involved: it would be dependent solely on the input source f-structure, in which case $W_s$ would need to be removed from any probability model.}

The components needed given (408), therefore, are (i) a source language LFG-DOP model, $P(R_s \mid W_s)$; (ii) the $\tau$ mapping (the translation model) plus the associated probabilities that a source f-structure produces a target equivalent, $P(R_t \mid R_s)$, both of which are calculated in the same way as for the main LFG-DOT1 model above.

### 6.3.2 Model 2: Translation via $\tau$ and $\gamma$

Recall that LFG-DOT Model 2 uses integrated bilingual LFG-DOP corpora, relating each $c$-structure node to its corresponding f-structure fragment (using LFG-DOP-$\phi$) and its corresponding target $c$-structure node (using $\gamma$ the DOT2 model). In addition, for each source f-structure fragment, we relate that to its corresponding fragment in the target f-structure (using $\tau$, the LFG transfer mechanism). While we ultimately reject LFG-DOT2 in favour of models which state the translation relation using only $\gamma$, we can contrast here the probability model (407) of LFG-DOT1 with the general LFG-DOP probability model (Bod & Kaplan, 1998), which computes probabilities for the full $(c,f)$ representations, and not for f-structures alone. Model 2, therefore, requires the use of their probability model, in which case (407) can be reduced to (409):

$$
P(W_t \mid W_s) = \sum R_{s,t} P(R_s \mid W_s)P(R_t \mid R_s)$$

$P(W_t \mid R_t)$ can be omitted since it is trivially equal to 1, since if $R_t$ is given, then the target string $W_t$ can always be seen, this being a component part of the target c-structure. We discussed in the previous section why such a simplification cannot be made in the LFG-DOT1 probability models.

Given this, it should be clear that $P(R_s \mid W_s) = P(R_s)$, so (409) is further reducible to (410):

$$
P(W_t \mid W_s) = \sum R_{s,t} P(R_s)P(R_t \mid R_s)$$

An alternative, still simpler formulation of (410) is (411):

$$
P(W_t \mid W_s) = \sum R_{s,t} P(W_t \mid W_s)$$

(411) is further reducible to (412) since $W_t$ and $W_s$ are in $R_t$ and $R_s$ respectively if we assume that $R_{s,t}$ are the full $(c,f)$ representation pairs for the source and target strings $W_s$ and $W_t$:

$$
\sum R_{s,t} P(R_{s,t})
$$

As usual in LFG-DOP, $P(R_s)$ and $P(R_t)$ are equal to the sum of the derivations which make up their representations, where the derivations are equal to the product of all linked LFG-DOT fragments which can be combined to form these derivations (218), p.116, and the probability of each linked LFG-DOT fragment
is calculated as its frequency relative to all linked fragments in the corpus as a whole (219). Note that none of these LFG-DOT models which employ the full \( c, LFG-DOP-\phi, f \) representation for strings require any additional Markov assumptions. The basic translation units are pairs of linked LFG-DOP fragments, and the basic stochastic event is the combination of two linked LFG-DOP fragment pairs.

It remains an open question as to whether any of the probability models offered here will give reasonable results for actual bilingual LFG-DOT corpora, as at the time of writing no such corpora exist. Nonetheless, (409)-(412) are simple and correct formulae on which LFG-DOT2 can be based. The probability of \( P(W_t | W_s) \) is computed by summing the probabilities of all \( \langle R_t, R_s \rangle \) pairs that generate \( W_t \) and \( W_s \), where the probability of each \( \langle R_t, R_s \rangle \) pair is computed as the sum of the probabilities of its derivation-pairs, where each derivation-pair is the product of its linked fragment-pairs, and where each linked fragment-pair has a probability equal to its normalized relative frequency, i.e. normalized by the fragments it \textit{competes} with (cf. (222)-(225), p.117f., and discussion thereon). Ultimately, of course, we need to choose the \( W_t \) for which \( P(W_t | W_s) \) is maximal.

Probability models (407) and (408), on the other hand, are quite different models where links exist only at the level of f-structure (using \( \tau \)). Again, we cannot say whether they would work effectively in practice, but if we have enough data there appears no strong reason to believe that they ought not to. In contrast to the Bod & Kaplan (1998) models, (407) and (408) show that LFG-DOT can also be treated in a similar manner to the speech-speech translation models of Chang & Su (1993, (404) above) and Alshawi (1996, (405) above). It may be safer to rely on the original LFG-DOP probability models as a basis for LFG-DOT, but only widespread experimentation on suitably large-scale corpora will tell definitively. Nevertheless, we have described why the LFG-DOP probability models of Bod & Kaplan (1998) cannot be used as a basis for either LFG-DOT1 model.

We described in section 2.2.1 how DOP was adapted to deal with the problem of unknown words. Given that this will also be a problem for LFG-DOT, we need to factor this into our probability models. One rather simple way to alleviate the sparse data problem somewhat would be to preprocess our examples with a morphological analyzer to lemmatize morphological variants of words. This is similar to the LTAG approach (Abeillé \textit{et al.}, 1990) where a word with the description \{\texttt{\textbackslash leaf}\} subsumes any of its morphological variants \{\texttt{\textbackslash leaf, left, leaves...}\}. We observed in section 6.2.3 that \textit{Discard} can be used to mimic any required lemmatization process. We advocated there the adaptation of the methodology put forward for Tree-DOP to LFG-DOP, and subsequently LFG-DOT. As we reported in section 2.2.2, DOP3 used the Good-Turing method (Good 1953) to adjust the observed probabilities of fragments to take into account the fact that no corpus, no matter how large, can expect to cover the total population of expected events. Adapting all of the probability models given here can be performed straightforwardly using the description of the Good-Turing method given in section 2.2.2. Finally, we show in section 6.3.6 that LFG-DOT probability models can also be adapted to incorporate the \textit{Discard} function of LFG-DOP, both for reasons of robustness as well as for lemmatization in an extended transfer stage, in order to overcome problems of limited compositionality.
6.3.3 Model 3: Translation via $\gamma$ with Monolingual Filtering

LFG-DOT3 eschews $\tau$-equations and states the translation relation solely in terms of $\gamma$. F-structures are used for monolingual filtering only. The probability model given in (410) for LFG-DOT2 can also be used for LFG-DOT models 3 and 4. That is, the probability of a target string $W_t$ given a source string $W_s$ can be calculated by multiplying the probability of the source representation $R_s$ by the probability of the target representation given the prior probability of the source representation. The only difference between models 2 and 3 (or 4) concerns the nature of the linguistic representations involved: in LFG-DOT2, the languages are linked via the DOT2 $\gamma$ link as well as via $\tau$, whereas only $\gamma$ is used as an interface in subsequent models.

If we adopt (410) as the probability model for LFG-DOT 3 (and 4), we require the following components:

- $P(R_s)$: the probability of the source structures, a full $\langle c, \text{LFG-DOP-$\phi$, $f$} \rangle$ representation;
- $P(R_t \mid R_s)$: transfer, in terms of the DOT2 $\gamma$ function;
- $P(R_t)$: the probability of the target structures, a full $\langle c, \text{LFG-DOP-$\phi$, $f$} \rangle$ representation.

The probability of the source representation $P(R_s)$ is calculated in the normal way (cf. (407) and subsequent discussion). The transfer component $P(R_t \mid R_s)$ can be estimated by $\frac{P(R_t \mid R_s)}{P(R_s)}$, i.e. dividing the probability of the linked source and target $\langle c, f \rangle$ structures $R_t$ and $R_s$, by the probability of the source representations $R_s$. Comparing (410) with (407), note again that a full LFG-DOP source representation contains the source string, so $P(R_s \mid W_s)$ is omissible without loss of information.

6.3.4 Model 4: Translation via $\gamma$ and ‘Extended Transfer’

The only difference between LFG-DOT models 3 and 4 is that a small amount of the probability space in LFG-DOT4 is given over to lemmatized translation pairs in the extended transfer phase. As we stated in section 6.2.4, we envisage that this be kept to a minimum via Good-Turing, but until large-scale corpora are available we cannot say to what extent this part of the probability space should be limited. As we show with respect to Discard in section 6.3.6, however, we are sure that unless some limit is placed on newly formed fragments, these latter will quickly take over from those fragments derived via regular LFG-DOP techniques to the detriment of the overall probabilities of sentences. In the context of translation models, this will lead to previously correct translations being less preferred than ‘ungrammatical’ variants. We discussed how Good-Turing techniques can be applied in section 2.2.2 with respect to DOP, and their application to LFG-DOT models is straightforward, as shown in section 6.3.6.

6.3.5 Other Possible Probability Models

The probability models presented in sections 6.3.1–6.3.4 are all based on the relative frequency of a fragment in a treebank. Certain problems have been pointed out in the literature with such probability models, both for DOP and LFG-DOP. These accounts are presented in this section. Large-scale testing of the LFG-DOT architectures and associated probability models may cause one of the following alternative fragment
estimators to be required. We shall examine the main points of each in turn. We summarize the LFG-DOT probability models foreseen in section 6.3.7.

Random Fields

Abney (1997) points out that while probabilistic versions of regular and context-free grammars (namely HMMs and PCFGs, respectively) have been around for some time and are well understood, “no satisfactory probabilistic analogue of attribute-value grammars has been proposed; previous attempts have failed to define a correct parameter-estimation algorithm”. As we stated in the introduction to chapter 4, one of the main reasons for using LFG is that it can cope with phenomena for which regular and context-free grammars are inadequate. Accordingly, this is why it was combined with DOP to form LFG-DOP (Bod & Kaplan, 1998; 1999), and also why we adopt LFG-DOP as a language model in our LFG-DOT translation models.

Previous efforts at probabilistic versions of attribute-value grammars (Eisele, 1994; Brew, 1995) associate weights with analogues of grammar rules (Horn clauses in the case of Eisele; typed feature structures in Brew’s case). Abney terms this the Expected Rule Frequency (ERF) method. Both approaches, however, fail to deal adequately with re-entrancies in feature structures. Given that this phenomenon expresses a context dependency, and Eisele’s proposal assumes independence in his model, it is not surprising that re-entrancies pose problems. Nevertheless, Abney points out that “solutions to the context-sensitivity problem have long been known ... [and] are known as random fields. [These] can be seen as a generalization of Markov chains and stochastic branching processes”.

Abney describes the problem that stochastic CFGs have with reference to the weighted grammar in (413):

\[
\begin{align*}
\text{a.} & \quad S \rightarrow A \ A \quad \beta_1 = 1/2 \\
\text{b.} & \quad S \rightarrow B \quad \beta_2 = 1/2 \\
\text{c.} & \quad A \rightarrow a \quad \beta_3 = 2/3 \\
\text{d.} & \quad A \rightarrow b \quad \beta_4 = 1/3 \\
\text{e.} & \quad B \rightarrow a \ a \quad \beta_5 = 1/2 \\
\text{f.} & \quad B \rightarrow b \ b \quad \beta_6 = 1/2 \\
\end{align*}
\]

The probability of a tree is equal to the product of the probabilities of the rules participating in the derivation of that tree. Therefore, the probability of tree (414) is \( \beta_1 \beta_3 \beta_3 = \frac{1}{2} \frac{2}{3} \frac{2}{3} = 2/9 \):

\[
(414) \quad \begin{array}{c}
S \\
\downarrow \quad \downarrow \\
A \quad A \\
\quad \downarrow \quad \downarrow \\
a \quad a
\end{array}
\]

Another analysis of the string \( aa \) is possible (413), namely \( s_b(aa) \). The probability of this parse is \( \beta_2 \beta_5 = \frac{1}{2} \frac{1}{2} = 1/4 \). Given that \( 1/4 > 2/9 \), this latter parse should be the preferred analysis of the string \( aa \).
Abney goes on to show that the estimation of parameters (i.e., estimating the weights \( \beta_i \)) cannot be accomplished by standard techniques. If the weights in a grammar such as (413) are incorrect, then the usefulness of that grammar is devalued. Let us assume that we have a training corpus (415) consisting of 12 trees of four possible types from grammar (413):

\[
\begin{array}{cccc}
  x_1 & x_2 & x_3 & x_4 \\
  \text{S} & \text{S} & \text{S} & \text{S} \\
  \text{A} & \text{A} & \text{A} & \text{B} \\
  a & a & b & a \\
  \text{c} = 4x & 2x & 3x & 3x \\
  \hat{p} = 4/12 & 2/12 & 3/12 & 3/12 \\
  \text{= 12} & & & \\
\end{array}
\]

\( \hat{p}(x_i) \) denotes the empirical distribution of the \( i^{th} \) tree, i.e., its relative frequency, in contrast to \( \hat{q}(x_i) \), its probability distribution, as given in (413). To illustrate the difference, \( \hat{p}(x_1) = 1/3 \) in (415), but we have already seen that \( \hat{q}(x_1) = 2/9 \). Of course, what has happened is that the empirical probabilities do not correspond to the real probabilities as there are two trees missing from (415) which are possible via (413), namely \( s(A(a)_A(b)) \) and \( s(A(b)_A(a)) \). The sum of the real probabilities of the four types of tree in (415) is 7/9, so the missing two trees account for the remainder of the probability mass. Given that we cannot expect our models to necessarily contain all possible observations, the closer the weights \( \hat{p}(x_i) \) come to the actual weights \( \hat{q}(x_i) \), the better. This can be measured by the Kullback-Leibler distance, defined as (416):

\[
D(\hat{p} \parallel q) = \sum_x \hat{p}(x) \ln \frac{\hat{p}(x)}{\hat{q}(x)}
\]

(416) shows the distance between \( \hat{p} \) and \( q \) as the log of the ratio of \( \hat{p}(x) \) to \( \hat{q}(x) \). The overall distance \( D \) is the average distance between \( \hat{p} \) and \( q \). As an example, for tree \( x_1 \) we get \( \hat{p}(x_1) \ln (\hat{p}(x_1)/\hat{q}(x_1)) = \frac{1}{12} \ln \frac{1}{3/9} = 0.14 \). If we sum these values for the four trees in (415), given the weights \( \beta \) in (413) and observed frequencies \( p \) in (415), we get the total distance \( D(\hat{p} \parallel q) = 0.32 \). If we assume a different set of weights \( \beta \), then if the distance \( D \) gets smaller, then the new set of weights can be considered a better fit for the real probabilities \( q \), and if \( D \) increases then the new weights are a worse fit. Given this, we can see that the Kullback-Leibler distance is actually a measure of dissimilarity.

Abney shows that the weights derived by ERF are provably the best possible for any given regular or context-free grammar, in that the relative frequency of a tree converges to its true probability (in ‘large enough’ corpora), but this is not the case for an attribute-value grammar. Given that CF-rules are independent of all others, any dependencies derived from them must be considered accidental. Nevertheless, there does appear to be an underlying dependency in (415), namely that where we have two \( A \)’s, both of these rewrite the same way. We can capture this constraint in an attribute-value grammar (417), which corresponds to (413) with the dependency included:
The language given by this grammar is the set of dags in (418):

<table>
<thead>
<tr>
<th></th>
<th>x₁</th>
<th>x₂</th>
<th>x₃</th>
<th>x₄</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>S</td>
<td></td>
<td></td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>a</td>
<td></td>
<td></td>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>b</td>
<td></td>
<td></td>
<td>b</td>
<td>b</td>
</tr>
</tbody>
</table>

The remaining task is to assign weights to the AVMs in (417). Brew and Eisele proceed upon the same lines as for CFGs, by assigning the weights in (413) to the ‘rules’ in (417). The probability of a tree is the product of the weights of the rules participating in its derivation, as before. So the probability of tree \( x₁ \) in (413), \( \bar{q}(x₁) = \beta₁ \cdot \beta₃ \cdot \beta₃ = \frac{1}{2} \cdot \frac{2}{3} \cdot \frac{2}{3} = 2/9 \). The problem, of course, with this approach is that we are using the probabilities in (413), which allow for two extra trees, to model our probabilities for the dags in (418), which constitute the entirety of the possible events. \( \bar{q}(x_i) \) is not a proper probability distribution: it is ‘unnormalized’ (hence the ‘\~\’).

The approaches of Brew and Eisele now normalize these probabilities \( \bar{q}(x_i) \) by dividing them by 7/9, the total probability for the dags in (418) to ensure the probabilities sum to 1. Thus we get the results in (419) via ERF:

\[
q(x) = \frac{x₁}{2/7} = \frac{x₂}{1/14} = \frac{x₃}{9/28} = \frac{x₄}{9/28}
\]

However, Abney shows that the ERF method converges to the wrong weights for AVGs. Assuming the same distribution for the dags in (418) as in (413), the best weights are shown in (420):

\[
\begin{align*}
[S \ A \ A] & \quad [S \ B] & \quad [A \ a \ a] & \quad [A \ b \ b] & \quad [B \ a \ a] & \quad [B \ b \ b] \\
\beta₁ & = & \beta₂ & = & \beta₃ & = & \beta₄ & = & \beta₅ & = & \beta₆ & = \\
\frac{2+2\sqrt{2}}{6+2\sqrt{2}} & = & \frac{3}{6+2\sqrt{2}} & = & \frac{\sqrt{2}}{1+\sqrt{2}} & = & \frac{1}{2} & = & \frac{1}{2}
\end{align*}
\]

No normalization is required for these weights, as it is easy to see that rules with the same left-hand sides sum to 1. Comparing the weights in (419) to those in (420), we see that \( D(\tilde{p} || q) = 0.07 \) for (419), but \( D(\tilde{p} || q) = 0 \) for (420).

Random fields, however, ensure that one’s probability model converges to the correct weights. Weights no longer need to sum to 1 for rules with the same left-hand side, nor do properties need to be identical to the rules of the grammar. While this enables the empirical distribution to be captured using fewer properties than was previously the case, as only as many properties are used as are needed to distinguish between trees.
that have different empirical probabilities, parameter estimation becomes more complex. As well as selecting the optimal weights, we need to associate these weights with a set of properties. Abney uses the method of (Della Pietra et al., 1997) for both these tasks. For property selection, the Kullback-Leibler divergence is measured presupposing the addition of each feature to the field, and the feature with the largest gain (i.e. biggest reduction in Kullback-Leibler distance) is added to the field. This process continues iteratively until no significant gain is realized. In the parameter estimation process, the parameters are estimated using an iterative scaling algorithm. Della Pietra et al. state that their method is an improvement on more generalized methods as there is no prerequisite that the features sum to a constant, and furthermore that larger steps are made towards the maximum at each iteration so that the rate of convergence may increase. Given that there are potentially many thousands of parameters to estimate, maximum entropy is used to temper the classic problem of overtraining (overfitting the parameters to the data), so that generalization to new configurations is permitted. Given also that the configuration space is potentially huge, Della Pietra et al. use Monte-Carlo sampling to estimate the expectations of random variables. A problem which is typical of such empirical approaches is that there may not be a sufficiently large enough occurrence of the desired features, in which case the estimates produced via Monte-Carlo sampling may be unreliable.

**Cormons’ Approach**

In contrast, Cormons (1999) states that the random field method has too great a complexity to be used in practice. Indeed, Johnson et al. (1999) state that “the Monte-Carlo parameter estimation procedure that Abney proposes seems to be computationally impractical for reasonable-sized grammars”.19 Cormons’ work is the first approach to actually include an implementation of LFG-DOP. He uses a DOP-based approach as despite it being “mathematically less justified than the approaches of Brew and Eisele, it nevertheless has the merit of taking into account a larger number of statistical dependencies. What is more, the complexity [of DOP] is less than Abney’s approach” (Cormons, 1999:5, our translation). He also makes the similar point to Abney (1997) that the ERF method does not make a Tree-DOP corpus ‘maximally likely’, despite several errors in his calculations (Cormons, 1999:31f.) . The same can be said for LFG-DOP (op cit., p.52).

In addition, Cormons shows that the LFG-DOP probability model M1 does not predict the correct sentence-analyses for a test set from the *Verbmobil* corpus. He gives three possible solutions as to how this model could be improved: (i) testing LFG-DOP with probability models M9 and M10; (ii) using a less pervasive fragmentation method; and (iii) using a probability model based on random fields.

How this latter tallies with his claim that the random field method is too complex to be used in practice is unclear. Furthermore, as none of these suggestions are actually taken on board it is impossible to say how well any of them would work empirically.20 Nevertheless, Cormons makes clear the reason why a probability model based on random fields is needed. All DOP models which have been published to date use leftmost substitution as the composition operator, and as we have seen LFG-DOP is no different. Given the added

19 This model has been used to rank translation candidates. On small-scale experiments, their models are able to identify the correct parse from the set of candidate analyses approximately 50% of the time. Given that it also gets it wrong approximately 50% of the time, it would seem that manual checking of the system’s selection is required 100% of the time.

20 The work of Van Genabith et al. (1999b) would appear to indicate that using probability models M9 and M10 is impractical given the large number of parses possible. See Table 4.6 and Table 4.7, p.159, and resultant discussion.
unification machinery of LFG-DOP, the selection of a fragment may restrict the subsequent choice of other fragments, depending on the probability model chosen.

For example, we showed in (230), p.119, that certain bona fide Tree-DOP derivations are ruled out in LFG-DOP models incorporating the LFG Uniqueness constraint. In addition, this is also partly a function of DOP's leftmost substitution stipulation on how fragments may be recombined. For example, we showed that the probability of derivation (259), p.134, for John walked differs depending on the probability model chosen: if the uniqueness check is not invoked, then John can be combined with a VP fragment with any number. If it is invoked, then only VPs with NUM=SG can combine, which may restrict the number of VPs in the competition set, resulting in different probabilities for the same derivation. Although it is omitted from his thesis, Cormons (personal communication) is convinced that using (say) rightmost combination may alter the probability models obtained using leftmost substitution, and given this assumption, Cormons (1999:84) recommends the use of random fields, stating that compared to DOP models, these produce probabilities without regard for how the representations are obtained. Nevertheless, we think this may not be an issue after all. The only way in which a SUBJ with NUM=SG can combine with a VP fragment derived from a VP with NUM=PL is via Discard, which by deleting one of these NUM features will enable unification. In models M0 and M1, the uniqueness check is not a barrier to such recombination, of course.

Finally, Cormons demonstrates (op cit., p.63) that if A1 is a more specific analysis of a sentence than A2, then A1 is more probable than A2. That is, the presence of an attribute is more probable than its absence. He states that "this is desirable ... as most features must be present for the parse to be correct" (ibid.). Nevertheless, one has to wonder whether this has a linguistic analogue: what would we expect to see more often—a man, or a man with red hair? Or a verb, or a tensed verb? We merely raise this as an issue for future consideration, it being outside the scope of this thesis.21

Probability in Terms of Occurrence in Derivations

Bonnema et al. (2000) point out a failing of the probability model of Tree-DOP, namely a preference for fragments derived from larger trees rather than those fragments which occur frequently. We described this in section 2.2.3 with reference to example ((70), p.67).

21Cormons (1999) performs an experiment in which Monica loves Bill is parsed as MALE loves FEMALE—only when Discard is used—despite there being an exact instance of Monica loves Bill with the correct feature values in the corpus. The reason for this is that it is outranked by other MALE loves FEMALE sequences from the corpus which causes the wrong analysis in this specific example. This again shows the potential of generalized fragments to have a negative impact on the analyses obtained. Cormons suggests that such behaviour might be corrected if a random field probability model were used. However, Bod (personal communication) thinks that this may not rectify Cormons' problem, "as the unlexicalized fragments are so numerous that their most frequent corresponding features will win". Only extensive testing will resolve this issue.
In the light of this undesirable quality, they propose a new probability model which addresses this concern. Given the original probability model of Tree-DOP, the more often we see a particular fragment the more evidence we have in favour of its occurrence. That is, it becomes more plausible. Bonnema et al. suggest that a better measure is to see how often a tree fragment is actually used in the treebank. That is, as well as its relative frequency, the evidence in favour of a fragment is the fraction of derivations that contain a substitution of that fragment (its fragment distribution). This is defined in (421) as \( \phi(\alpha, \tau) \), the fraction of all derivations of \( \tau \) that start with \( \alpha \). If \( \delta(\tau) \) denotes the set of all possible derivations of a constituent \( \tau \), then:

\[
\phi(\alpha, \tau) = \frac{|\{d_j \in \delta(\tau) : \alpha_{ij} \circ \ldots \circ \alpha_{kj} = \tau\}|}{|\delta(\tau)|}
\]

with \( \alpha_{ij} = \alpha \).

The prior probability that a fragment \( \alpha \) is used in the derivation of a constituent \( \tau \) is then calculated for each \( \tau \) having the same root symbol as \( \alpha \), multiplied by the probability that \( \tau \) is selected from the treebank. In order to compute the substitution probability of a fragment with respect to the treebank, this product is summed over the set of all constituents in the treebank, as in (422):

\[
p'(\alpha) = \sum_{\tau \in C_\alpha} F(\tau) \phi(\alpha, \tau)
\]

Bonnema et al. go on to show that the probability of a parse tree can be viewed as the average of the products of the relative frequencies of the fragments involved in each possible derivation of the parse tree.

This new DOP model is then compared to PCFGs as well as the original Tree-DOP model. The new DOP model is identical to a PCFG when the independence assumption is validated by the data. Furthermore, the new DOP model improves on a PCFG when the independence assumption is not validated. The corpus they use (their Figure 2) is reproduced here as (423):

\[
\begin{align*}
(423) & \begin{array}{cccccccc}
S & S & S & S & S & S & S & S \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
\end{array}
\end{align*}
\]

PCFGs (like CFGs) assume that all rules are independent of all others. (423) is equivalent to the PCFG \( G \) in (424):

\[
\begin{align*}
S & \rightarrow A & [0.25] & A & \rightarrow 0 & [0.5] \\
S & \rightarrow B & [0.25] & A & \rightarrow 1 & [0.5] \\
S & \rightarrow AB & [0.5] & B & \rightarrow 0 & [0.5] \\
& & B & \rightarrow 1 & [0.5]
\end{align*}
\]

The relative frequencies imposed upon the trees by \( G \) and the independence constraint is given in Table 6.1. The PCFG, employing the independence constraint, assigns correct probabilities to the trees in (423)—this can easily be checked by using the rules in (424). The new DOP model also gives the correct results.
<table>
<thead>
<tr>
<th></th>
<th>(a)</th>
<th>(b)</th>
<th>(c)</th>
<th>(d)</th>
<th>(e)</th>
<th>(f)</th>
<th>(g)</th>
<th>(h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rel. freq. of tree</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
</tr>
<tr>
<td>PCFG</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
</tr>
<tr>
<td>New DOP</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
<td>1/8</td>
</tr>
<tr>
<td>Former DOP</td>
<td>1/12</td>
<td>1/12</td>
<td>1/12</td>
<td>1/12</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
<td>1/6</td>
</tr>
</tbody>
</table>

Table 6.1: Relative Frequencies and Probabilities of the Trees in (423)

Bonnema et al. show that under the independence constraint the new DOP model is always equivalent to a PCFG. As we showed earlier, this is useful in that polynomial algorithms such as (Younger, 1967; Earley, 1970) can be used for parsing. Table 6.1 also shows that the original Tree-DOP model assigns twice as high a probability to binary branching trees compared to unary branching ones, and these figures are clearly wrong. Bonnema et al. then conduct a second experiment to show that when the independence constraint is not validated by the data, their new DOP model improves upon PCFGs. The new treebank replaces (423a) and replaces the \( A(0) \) in (423e) with \( A(1) \). The outcome of this is to make (a) and (b), and (e) and (g), identical, so our new corpus consists of the 6 trees in (425):

\[
\begin{array}{cccccccc}
S & S & S \\
A & B & B \\
0 & 0 & 1 \\
\end{array} \quad \begin{array}{cccccccc}
S & S & S \\
A & B & A \\
1 & 0 & 1 \\
\end{array} \quad \begin{array}{cccccccc}
S & S \\
A & B \\
0 & 1 \\
\end{array} \quad \begin{array}{cccccccc}
S & S \\
A & B \\
1 & 1 \\
\end{array}
\]

(425)

The PCFG is identical to (424), as the relative frequencies of the application of the rules has not changed. The relative frequencies of the new trees in (425) are shown in Table 6.2. Bonnema et al. observe that their DOP model accurately reflects the dependencies in the trees. There are four trees with probability 1/8: (b), (c), (e) and (f). Tree (e) has a clear internal dependency, namely that \( S(AB) \) and \( S(0) \) tend to avoid each other, whereas tree (f) has the opposite dependency. Their probabilities are respectively 1/64 below and above the probabilities for the PCFG. Trees (b) and (c) have no internal dependency: if \( S(B) \) is selected, this has no bearing on the probability of \( B \) being either 0 or 1. Thus their probabilities are the same as in the PCFG model (1/8), exactly between the probabilities for (e) and (f). The original DOP model does manage to show the internal dependencies, but with even greater differences related to any difference in size: trees (d), (e) and (f) all get the highest probabilities, which are not justified by the data.

In sum, the new DOP model of Bonnema et al. corrects the counter-intuitive predictions of the original DOP model.\(^{22}\) Rather than fragment size being of primary importance, a fragment’s overall occurrence frequency now determines its plausibility. Despite these improvements of their new DOP model over the built-in bias,

\(^{22}\)However, the original, biased relative frequency estimator still outperforms the estimator of Bonnema et al. on the WSJ. This is still under investigation (Bod, personal communication).

As an alternative, Bod (2000a) works out a fragment estimator based on maximum likelihood training using an EM algorithm. This fragment estimator is statistically consistent, and furthermore improves over both DOP and a trigram model on the OVIS corpus with respect to word error rate.

241
of the original model, however, the most probable parse (i.e. a disambiguation process) still cannot be solved by polynomial algorithms (cf. Sima’an 1996; 1999) using their new DOP model.

### 6.3.6 Incorporating Discard into the LFG-DOT Probability Models

Bod (2000a) presents an empirical assessment of LFG-DOP (Bod & Kaplan, 1998) and suggests an improvement over the techniques employed therein. The original LFG-DOP model treated all fragments as probabilistically equal regardless of whether these fragments contained generalized features, i.e. whether they were produced by Discard or not. Bod (2000a) proposes instead to treat generalized fragments as unseen events (using Good-Turing), and assign them probabilities by discounting. That is, the probabilities of known events are discounted (reduced) and the newly available probability mass is distributed among the unseen events. Similar to Way (1999) (cf. also section 4.5), Bod creates two separate bags of fragments—one including fragments derived via Root and Frontier, and the other containing fragments derived via Discard.\(^{23}\)

Of course, the Discard bag are not ‘unseen’ events, so using Good-Turing might be deemed inappropriate. Nevertheless, they are in effect ‘hidden’ events, in that they do affect the probabilities of ordinary Discard-free derivations. The point is this: in LFG-DOP (Bod & Kaplan, 1998) the Discard fragments occupy an unjustifiably large proportion of the probability space. We demonstrate this in the next few sections.

**Impact on Corpus Size in LFG-DOT1**

As an example, we computed the number of LFG-DOP fragments generated by Discard for the sentences in (334), p.179, with the original S1 replaced by the variant in (374). The results are shown in Table 6.3.

We show how the figures for the French sentences (on the right-hand side of Table 6.3) are calculated in Table 6.4. Table 5.1 shows that the total number of monolingual DOT fragments for the French sentences is 337. Where the Discard operation is excluded, the number of LFG-DOT1 fragments remains the same as the equivalent DOT corpus, as for each c-structure pairing there is one adjoining f-structure pair (only the English S1 has changed). Where Discard is applied, the c-structure to f-structure relationship becomes

\(^{23}\)Given that DOP is a performance model with claims to be ‘psychologically real’, it is an interesting question as to whether ill-formed fragments in the IFB should be moved to the WFB as soon as they are encountered. One piece of anecdotal evidence in favour of such a move is that when I first came to Ireland, I did not understand the ‘after X-ing’ (meaning ‘have just X-ed’) in Hiberno-English, and adjudged such strings as ungrammatical. After prolonged exposure to such utterances, they became grammatical to me, and after an even longer period, I actually produced such strings (despite myself)! This shows that one’s language models need to be dynamic.

---

Table 6.2: Relative Frequencies and Probabilities of the Trees in (425)
Table 6.3: The Effect of Discard on Monolingual LFG-DOP Treebanks

<table>
<thead>
<tr>
<th>English Fragments after Discard</th>
<th>French Fragments after Discard</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>N</td>
</tr>
<tr>
<td>S1</td>
<td>128</td>
</tr>
<tr>
<td>S2</td>
<td>128</td>
</tr>
<tr>
<td>S3</td>
<td>128</td>
</tr>
<tr>
<td>S4</td>
<td>16</td>
</tr>
<tr>
<td>S5</td>
<td>80</td>
</tr>
<tr>
<td>S6</td>
<td>128</td>
</tr>
<tr>
<td>S7</td>
<td>128</td>
</tr>
<tr>
<td>S8</td>
<td>128</td>
</tr>
<tr>
<td>S9</td>
<td>128</td>
</tr>
<tr>
<td>S10</td>
<td>128</td>
</tr>
<tr>
<td>Total</td>
<td>120</td>
</tr>
</tbody>
</table>

Table 6.4: Detailed Breakdown of the Discard Fragments in a French LFG-DOP Treebank

<table>
<thead>
<tr>
<th>Categories of Fragments</th>
<th>Sentence</th>
<th>NP</th>
<th>VP</th>
<th>V</th>
<th>Adj</th>
<th>Det</th>
<th>N</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>S¹</td>
<td>S²</td>
<td>S³</td>
<td>S⁴</td>
<td>S⁵</td>
<td>S⁷</td>
<td>S⁸</td>
<td>S¹⁰</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>12</td>
<td>10</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>12</td>
<td>10</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>12</td>
<td>10</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>12</td>
<td>10</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>12</td>
<td>10</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>12</td>
<td>10</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>Subtotal</td>
<td>80</td>
<td>90</td>
<td>14</td>
<td>41</td>
<td>37</td>
<td>37</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>Total</td>
<td>191</td>
<td>41</td>
<td>78</td>
<td>10</td>
<td>1</td>
<td>8</td>
<td>8</td>
<td>337</td>
</tr>
<tr>
<td>Sum</td>
<td>4136</td>
<td>164</td>
<td>1800</td>
<td>272</td>
<td>4</td>
<td>32</td>
<td>32</td>
<td>6440</td>
</tr>
</tbody>
</table>

eone to many, of course. We explained in section 4.1.1 why the number of LFG-DOP fragments following the application of the Discard operation is $2^{n+1}$ in the general case where $n$ is the number of features. As an example, consider the French sentence in (374), *Son suicide est tragique* (i.e. S1 here). In Table 5.1, we see that before Discard, the number of DOT c-structures (and, by extension, f-structures for LFG-DOP) for category S was 25. In Table 6.4, we see that of these 25 sentential f-structures, 10 contained 5 discardable features (hence $2^5$), 9 contained 4 discardable features (hence $2^4$), and 6 contained 2 discardable features (hence $2^2$). Summing these, we see that 10+9+6=25, the number of original DOT c-structure fragments in Table 5.1. Summing the $2^n$ elements we get $(10 \times 2^5)+(9 \times 2^4)+(6 \times 2^2) = 488$, the number of s-fragments for the French sentence S1 as seen in Table 6.3.

The last two rows in Table 6.4 sum up quite dramatically the effect of Discard. The number of s-fragments has increased from 191 DOT (and hence LFG-DOP, no Discard) fragments to 4136 LFG-DOP (with Discard) fragments (21.7 times as many fragments), NPs show an increase from 41 to 164 (4 times as many—we showed earlier with respect to Table 4.2, p.126, that NPs increase at a rate of $2^{#NP_{-feature}}$), VPs from 78 to 1800 (23 times as many), verbs from 10 to 272 (27 times as many), and the other categories all show
an 8-fold increase. The reason for the huge increase in S, VP and V fragments is due to equations such as 
\( \phi(n1) = \phi(n3) = \phi(n4) \) in (210), p.112), which show that the outermost root f-structure is associated with all three inextricably linked categories.

As Bod (2000a) discovers, the Discard fragments occupy an unjustifiably large proportion of the probability space: in Table 6.3, they account for almost 95% of the probability space for the French sentences, and just over 81% of the probability space for the English strings. However, using Bod’s (op cit.) technique to assign a fixed, small probability mass to the fragments generated by Discard via Good-Turing will ensure that the derivations using the ‘good’ non-Discard fragments will still be favoured. Given that, the probabilities of our translations in (376), p.208, will remain largely unaltered.

The French f-structures contain more discardable features than the English equivalents. Again, to take an example for S1 (374), p.208, the monolingual LFG-DOP representations are shown in (426):

\[
\begin{array}{c}
\text{S} \\
\text{NP} \quad \text{VP} \\
\text{Det} \quad \text{N} \quad \text{V} \quad \text{Adj} \\
\text{His} \quad \text{suicide} \quad \text{is} \quad \text{tragic} \\
\end{array}
\]

\[
\begin{array}{c}
\text{S} \\
\text{NP} \quad \text{VP} \\
\text{Det} \quad \text{N} \quad \text{V} \quad \text{Adj} \\
\text{Son} \quad \text{suicide} \quad \text{est} \quad \text{tragique} \\
\end{array}
\]

It can be seen that the French f-structure contains an extra feature GEND=MASC.24 On the French side, there are two discardable features in the outer SUBJ (NUM and GEND), as well as TENSE, plus the re-entrant SUBJ features in the XCOMP. There are, therefore, 2⁵ f-structure fragments derivable from the French f-structure in (426), so this pair would constitute one of the 2⁶ S-fragments for (374) in Table 6.4.

The impact of this small difference is striking. We saw in Table 5.1, p.180, that the number of fragments for the sentences in (334) was very nearly the same for both English and French (338 vs. 337, respectively), which one might expect given the similarities between the languages. The number of LFG-DOP fragments for each language would be exactly the same as the number of DOT fragments if the Discard operation is excluded. Nevertheless, the number of fragments when Discard is allowed free rein as in the model of (Bod & Kaplan, 1998) causes the English and French treebanks to be quite different in size, as Table 6.3 shows.

Apart from a relatively small number of differences in S1 and S5, the main contributor to the vast disparity in the number of fragments in Table 6.3 is the appearance of this extra NP feature (GEND) in the French examples. Despite this seemingly small difference, the effect on the French LFG-DOP treebank is clear, as

\[\text{24 Many linguists would balk at the analysis of the possessive pronouns in (426) as SPECS. We have, however, maintained this analysis here to prevent further explosion of the number of fragments derived via Discard.}\]

24
it is over 3.6 times the size of the corresponding English treebank. As we mentioned in section 4.1.1, it is not beyond the bounds of probability that still more features will be required, resulting in an exponential increase in the number of fragments (cf. Table 4.2, p.126, and surrounding discussion).

**Impact on Corpus Size in LFG-DOT2**

In Table 5.1 (p.180), we gave the number of DOT fragments for the sentences in (334). For the slightly different treebank where we replace SI in (334), namely Le suicide est tragique $\leftrightarrow$ Suicide is tragic, by Son suicide est tragique $\leftrightarrow$ His suicide is tragic in (374), the number of linked DOT fragments equals 336. Table 6.3 showed the number of LFG-DOP fragments for the monolingual treebanks built from these strings. As we have just stated, the number of French fragments outnumbers their English counterparts except for S4. The maximum number of possible linked LFG-DOT2 fragments, therefore, can be calculated by adding the 76 extra English fragments for S4 to the French total, and subtracting all the differences in number between each sentence. This is because an LFG-DOT2 corpus permits only a one to one linking between source and target fragments. Performing this calculation, we thus arrive at a potential number of 1726 linked fragments. Given some of the complex examples discussed in this thesis, there will, however, be certain source fragments which cannot be linked to a target equivalent (cf. (336) and (347), p.181, for example). Table 6.5 shows the number of LFG-DOT fragments before and after the application of Discard, once the impossible links have been rejected.

In fact, of the 1726 candidates, only the 8 V-fragments from the French fragments for S5, the commit suicide $\leftrightarrow$ se suicider example, are rejected as there is no English verbal link possible (they link only at VP, not at V, as (337) showed). All other rows in the right-hand side of Table 6.5 are filled with the numbers for the English LFG-DOP fragments from Table 6.3. As these are smaller in number than their French counterparts, there can only be this number of bilingual LFG-DOT links, as the relationship is one to one.

Once more, we see the effect of Discard on the number of eligible fragments, but this increase is not as dramatic as in the monolingual LFG-DOP treebanks in Table 6.3. We see in Table 6.5 an increase in s-fragments from 191 to 1072 (5.6 times as many), 460 vp-fragments compared to 78 (5.9 times more fragments), 59 more v-fragments (up 7.6 times), and double the number of fragments for all other categories. We see that
the fragments created by Discard occupy just over 80% of the probability space. We propose to use Good-
Turing to reduce this proportion significantly to ensure that the fragments created by Root and Frontier
continue to have the greatest effect in terms of the probability of derivations of candidate translations.

Impact of Lexicalization on Corpus Size in LFG-DOT1

Most experiments with DOP-based models prune the search space, usually by restricting (a combination
of) the depth of fragments, the number of lexicalized fragments and/or the number of substitution sites per
fragment. If we take the treebank derived from the sentences in (334), p.179, and impose restrictions on
the minimum amount of lexicalized fragments per fragment, then the number of total fragments is shown
in Table 6.6. Lexicalization >= 1 means that all fragments containing no lexical items are removed from
the treebank, and Lexicalization >= 2 implies that at least one of the languages has two (or more) lexical items
in each ⟨source,target⟩ paired fragment. Recall that 321 fragment pairs participate in the LFG-DOT corpus.
We can see that stipulating that there must be at least one lexical item causes 16.5% of the original corpus
to be deleted, and if we raise this to two lexical items then 49.6% of the original fragments are removed.

Lexicalization also has a significant effect on the translation process. For our translations in (339), p.181,
the probabilities of each of the translations with respect to the corpus when lexicalization >= 1 is shown in
(427):^25

(427) a. P(John commits suicide ← Jean se suicide) = 0.00096 (≈ 1/1071, previously 1/1576)
    b. P(John commits suicide ← Jean commet le suicide) = 0.00148 (≈ 1/676, previously 1/1238)
    c. P(Mary commits suicide ← Marie se suicide) = 0.007539 (≈ 1/1337, previously 1/169)
    d. P(Mary commits suicide ← Marie commet le suicide) = 0.00148 (≈ 1/676, previously 1/1238)

Ranking these translations with respect to each other gives the results in (428):

---

^25Although we are comparing DOT2 translation probabilities with LFG-DOT1 probabilities, if Discard is omitted, and
depending on the competition set in our LFG-DOP probability models, the probabilities may be identical, as here, given that
for each DOT2 fragment there is one accompanying f-structure fragment.
(428) a. \(P(\text{John commits suicide} \leftarrow \text{Jean se suicide}) = \frac{96}{244} = 0.393\)
b. \(P(\text{John commits suicide} \leftarrow \text{Jean commet le suicide}) = \frac{148}{244} = 0.607\)
c. \(P(\text{Mary commits suicide} \leftarrow \text{Marie se suicide}) = \frac{7539}{9019} = 0.836\)
d. \(P(\text{Mary commits suicide} \leftarrow \text{Marie commet le suicide}) = \frac{1480}{9019} = 0.164\)

For *John commits suicide*, we see that the wrong compositional translation is preferred about 1.5 times more than the correct specific translation, and for *Mary commits suicide* the specific translation is preferred about 5.2 times as much as the default translation. Little has changed in this respect from the results in (341).

When lexicalization is set to at least two terminal items, we obtain the results in (429):

(429) a. \(P(\text{John commits suicide} \leftrightarrow \text{Jean se suicide}): \text{impossible to derive}\)
b. \(P(\text{John commits suicide} \leftrightarrow \text{Jean commet le suicide}) = 0.0038 \simeq \frac{1}{354}\)
c. \(P(\text{Mary commits suicide} \leftrightarrow \text{Marie se suicide}) = 0.014548 \simeq \frac{1}{69}\)
d. \(P(\text{Mary commits suicide} \leftrightarrow \text{Marie commet le suicide}) = 0.0038 \simeq \frac{1}{354}\)

The only way in which *Mary commits suicide* \(\leftrightarrow\) *Marie se suicide* is derivable under the stipulation that there be at least two lexical items per fragment is via the complete sentential tree, which contains two lexical items. Figure 6.1 shows the three derivations of *John commits suicide*. (337), p.181, shows the DOT linked pair for *commits suicide* \(\leftrightarrow\) *se suicide*. For each of the three possible prior derivations of *John commits suicide* \(\leftrightarrow\) *Jean se suicide*, each French S-fragment is excluded from the recombination process as none of them have at least two lexical items: the first is \(s(NP(Jean),VP)\), the second \(s(NP,VP)\) and the third \(s(NP,VP(v(se - suicide)))\). Therefore, with lexicalization set at 2, this translation is no longer achievable. As for *Mary commits suicide*, ranking the two translations gives the result in (430):

(430) a. \(P(\text{Mary commits suicide} \leftarrow \text{Marie se suicide}) = \frac{14548}{18348} = 0.793\)
b. \(P(\text{Mary commits suicide} \leftarrow \text{Marie commet le suicide}) = \frac{3800}{18348} = 0.207\)

The correct translation achieved via the specific \(\tau\)-equations is preferred about 3.8 times more than the wrong, compositional translation.

Finally, while acknowledging that these treebanks are small, the effect of lexicalizing the grammar as a pruning device is quite clear. The removal of certain fragments from the treebank increases the probabilities of the candidate translations with respect to the corpus as fewer translations are possible. However, it does not alter the probability of the translations with respect to one another to any great extent. (429) demonstrates that care must be taken when lexicalizing fragments, as blindly removing those which do not conform to the current requirements may result in translations no longer being derivable. This result must cast at least a degree of doubt on the process of automatically pruning the set of candidate fragments, which Sima’an states is required to make DOP-based experiments possible, as it seems to imply that some manual intervention may be necessary to ensure that certain translations remain possible. However, the danger of *ad hoc* procedures such as this is that the frequencies (and hence probabilities) of certain fragments will be altered in their favour, resulting in a set of skewed translation probabilities. The same criticism applies to arbitrarily deleting fragments in terms of their depth.
<table>
<thead>
<tr>
<th></th>
<th>Exact Match</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+Discard</td>
<td>-Discard</td>
<td>+Discard</td>
</tr>
<tr>
<td>Bod &amp; Kaplan 98</td>
<td>1.1%</td>
<td>35.2%</td>
<td>13.8%</td>
</tr>
<tr>
<td>Bod 2000a</td>
<td>35.9%</td>
<td>35.2%</td>
<td>77.5%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Exact Match</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+Discard</td>
<td>-Discard</td>
<td>+Discard</td>
</tr>
<tr>
<td>Bod &amp; Kaplan 98</td>
<td>2.7%</td>
<td>37.9%</td>
<td>17.1%</td>
</tr>
<tr>
<td>Bod 2000a</td>
<td>38.4%</td>
<td>37.9%</td>
<td>80.0%</td>
</tr>
</tbody>
</table>

Table 6.7: The Effect of the Discard operation on the Verb mobil and Homecentre corpora

Allocating a Fixed Probability Mass to Discard Fragments

Bod’s new probability model for LFG-DOP (Bod, 2000a) assigns a fixed probability mass to the Discard bag so that the exponential explosion of these fragments does not affect the probabilities of non-Discard fragments.

Bod (op cit.) performs an experiment on the Verb mobil and Homecentre corpora. He ignores the problem of unknown words by ensuring that all words in the test set are in the training set. Given also that the main aim was to investigate the impact of the Discard fragments, two sets of fragments were created: one containing all fragments (up to depth 4), and the other excluding all fragments derived via Discard. This makes the latter set of fragments equal to the LFG-DOP model of (Bod & Kaplan, 1998). His results are shown in Table 6.7. It is clear that if all fragments are used, then (Bod & Kaplan, 1998) performs poorly: exact matches are only 1.1% and 2.7%, against 35.9% and 38.4% by (Bod, 2000a), whilst Precision measures of 13.8% and 17.1% and Recall measures of 11.5% and 15.5% considerably underperform (Bod, 2000a)’s Precision figures of 77.5% and 80% and Recall 76.4% and 78.6%. Furthermore, Bod discovered that his new model always generates the correct analyses, whereas that of (Bod & Kaplan, 1998) does not. Perhaps more significantly, he observes that the LFG-DOP model of (Bod & Kaplan, 1998) performs better without Discard generated fragments! Bod notes that “this suggests that treating generalized fragments in the same way as ungeneralized fragments is harmful” (Bod, 2000a).

Note also that adding Discard generated fragments to the model of (op cit.) leads only to slight performance increases, none of which are significant when evaluated with paired t-tests. This suggests perhaps that on the small corpora used here, there are too many fragments to be reliably estimated. A more damaging suggestion might be that Discard generated fragments do not contribute significantly to the parse accuracy. However, this should be tempered slightly by recalling that one way in which Discard adds robustness to LFG-DOP models is in helping to deal with unknown words. As Bod has explicitly prevented unknown words in his corpora, any positive contribution by Discard in terms of the results in Table 6.7 is likely to be diluted.
6.3.7 What Probability Model do we need?

We presented in section 6.3 a number of possible probability models for LFG-DOT. Each of these are considered to represent reasonable probability models which will enable the LFG-DOT architectures presented in section 6.2 to work in practice. Each probability model is based on the relative frequency of a fragment in a treebank.

Certain researchers have pointed out a number of problems with such probability models. Abney (1997) states that assigning the weights of fragments in probabilistic AV-grammars is problematic, given the context sensitivities (re-entrancies) in the representations, unless probability models based on random fields are used. Cormons (1999) states that no practical computational procedures exist which enable the estimation of such parameters for grammars of a reasonable size. He also fears that probability models obtained using DOP’s leftmost substitution stipulation can bias probabilities assigned to linguistic objects. Bonnema et al. (2000) show a problem with DOP’s preference for larger trees, so that the ‘best’ analyses are not always the most preferred ones. Way (1999) advocates placing a limitation on the scope of Discard in LFG-DOT in order to minimize the negative effect of such fragments on LFG-DOT probability models. Bod (2000a) makes the same recommendation after testing a number of sentences on two different corpora.

For the small-scale experiments performed in this thesis, all probabilities are calculated using relative frequency. Given the problems with this approach described above, it may turn out that different probability models are required, which will alter the results accordingly. Only empirical testing on a much wider scale will show definitively whether this is required.

For the moment, therefore, our probability models remain premised on the notion of relative frequency. Although we do not test for it here, we anticipate adopting the Good-Turing method given in section 2.2.2 to deal with unknown words (and unknown category words) in our test corpora. Bod (2000a) employs Good-Turing also to account for the fragments in the Discard bag, so we propose that the probability models in section 6.3 be adapted using Good-Turing to deal with unknown fragment pairs. Formula (431) gives the probability of fragments produced via Root or Frontier:

$$P(f \mid f \text{ is generated by Root/Frontier}) = \left(1 - \frac{n_1}{N}\right) \frac{\# f}{\sum_{f' : f' \text{ is generated by Root/Frontier}} \# f'}$$

(432) gives the probability of fragments produced via Discard:

$$P(f \mid f \text{ is generated by Discard}) = \left(\frac{n_1}{N}\right) \frac{\# f}{\sum_{f' : f' \text{ is generated by Discard}} \# f'}$$

As before, Good-Turing estimates the probability mass of unseen events as \(\frac{n_1}{N}\) where \(n_1\) is the number of once-occurring events (‘singletons’) and \(N\) is the total number of observed events. This small fixed amount of the probability space is assigned to those fragments produced by Discard, while the remainder of the probability space is given over to the grammatically sound fragments derived via Root and Frontier.

Finally, LFG-DOT4 assumes an extra level of processing called ‘extended transfer’, in which Discard is used again to produce more general fragments from those produced in LFG-DOT3, which may be of more use in the search for the ‘best’ translation. We propose in section 6.3.4 that Good-Turing may be used once more to limit the amount of the probability space available to lemmatized translation pairs.

249
It is apposite here to note a final finding by Bod (2000c) in a separate experiment, namely that the functional structures of LFG-DOP help improve the parse accuracy of tree-structures to a statistically significant degree. Whilst we have been unable to rigorously test LFG-DOT on large-scale corpora, these being unavailable and too unwieldy to be constructed manually, we can take encouragement from this finding that LFG-DOT ought to outperform DOT, as is the tenet of this thesis.

6.3.8 What would be involved in building an LFG-DOT MT System?

Although we have left the building and testing of the LFG-DOT systems described here for further work, it is worthwhile considering what would be involved in building prototypes of each system. Of course, such systems could be built from scratch, but there may be resources already available which might be adapted for use in the proposed systems.

The descriptions of the LFG-DOT models in section 6.2 show that each model requires source and target LFG-DOP language models, and additionally LFG-DOT1 and LFG-DOT2 require \( \tau \)-equations to link elements of the source and target f-structures. The most obvious starting point would be the English-German section of the Verbmobil corpus. Both Cormons (1999) and Poutsma (2000) use this corpus for their experiments on LFG-DOP and DOT respectively. The LFG-DOP version of the corpus used by Cormons should be usable immediately for models LFG-DOT3 and LFG-DOT4. For the other two models, \( \tau \)-equations would need to be added. These could be added manually, or preferably by automatic means using the definition of \( \tau \)-support in (366).

What could we expect from such testing? Firstly, direct comparisons could be drawn with Poutsma’s (2000) experiments on DOT, so we could see quite quickly whether the anticipated benefits of the richer LFG-DOT models are achievable in practice. Secondly, such experiments would also inform the discussions on efficiency. We would be able to comment on the implications for time and space, and should these be too onerous, experiments could be performed which prune the search space, following the suggestions made in sections 4.4-4.6, such as using lexicalized fragments or limiting fragment depth. Furthermore, such testing would provide insight into the probability models proposed in section 6.3, especially where Discard is concerned. Nevertheless, given the two examples cited in (Poutsma, 2000; cf. (322)), neither of which contains a correct English sentence, one wonders how reliable any results might be.

Such experiments may well provide insights into two of the remaining problems for MT discussed in section 1.3, ambiguity and the subset problem. We suppose that the Verbmobil corpus contains sentences which are ambiguous either monolingually or in translation, or both. Furthermore, such investigation would enable us to state definitively whether any target structures produced by the LFG-DOT systems would indeed be acceptable to the target grammars. As to the third problem, namely the combination of exceptions, we have reason to doubt whether the Verbmobil corpus would be useful here, given the sublanguage involved. Telephone conversations involve relatively short sentences, so we hypothesize that there will be few, if any combinations of ‘hard’ translation cases in the corpus.

How might this be tested, then? One way forward might be to examine the Penn Treebank, for instance, for instances of difficult translational phenomena. As shown in this thesis, there are problems when such cases interact with straightforward examples as well as when they co-occur with other difficult instances.
Depending on what is the object of testing, we suppose that subsets of sentences could be extracted from the treebank as input into the next stage. The techniques described in section 4.7 could then be run over these treebank trees to derive source (English) f-structures. We would then need to link elements from the treebank trees with their corresponding f-structure fragments using the source language LFG-DOP model.

The extracted English strings would need to be translated into appropriate languages to test these difficult cases: some are problematic when translating from English to German, for instance, but others may only prove difficult when translated into (say) French. Once this has been done, these foreign language strings would need to be tagged and parsed, so that we derive a bilingual treebank. The newly created target language treebanks could again be subject to the techniques described in section 4.7, so that source and target LFG-DOP language models are derived.

This may be regarded as implausible; much work on information extraction, for example, is being performed using comparative corpora, such as newspaper texts. In this way contrastive issues can be studied, but as the source and target texts are unlikely to be translations of each other, the aligned corpora required for statistical approaches to MT would not be at hand. Construction of monolingual treebanks is non-trivial (cf. Marcus et al., 1993), but there is long term kudos to be gained—there is little doubt that the Penn Treebank is the de facto standard treebank used by many researchers in the area of statistical NLP. We can anticipate that one day there will be bilingual treebanks available, and it is to be expected that the developers of such resources will reap their reward. The alternative to translating a currently existing treebank is to take a bilingual corpus such as the Canadian Hansards and to tag and parse them instead. This too would take considerable resources, but is an undertaking worthy of consideration by corpus linguists.

Assuming that bilingual LFG-DOP corpora are indeed derivable, we need to map elements of the source language to their target counterparts. This could be done by hand, but might also be achievable automatically: Poutsma (2000) added semantic annotations to the trees of his corpus, thereby enabling elements of the source trees to be linked to their counterparts in the target language. It may be possible to do something similar for LFG-DOP. Given that the γ function in LFG-DOT models is imported from DOT2, it can be seen that models LFG-DOT3 and LFG-DOT4 could readily be tested. For the LFG-DOT models which require linkings between (source, target) f-structures, this could be automated using the definition of τ-support in (366).

As for LFG resources, obviously the best known example is the Xerox Linguistic Environment (XLE) system, a reimplementation of the LFG Grammar Writer’s Workbench (Kaplan & Maxwell, 1996). The goal of the ParGram project is to produce parallel grammars which cover large grammar fragments of different languages with sizeable lexicons. Furthermore, parsing of 40 word sentences is achievable in reasonable time. One could envisage using the parallel grammars of the XLE to produce source and target (c, φ, f) structures which could then be input into an LFG-DOP language model to produce fragments with associated probabilities. These could then be adapted in a similar manner as discussed above for LFG-DOT.

In sum, while a considerable amount of effort would be required in order to develop sizeable LFG-DOT fragments, there are a number of possible ways ahead to achieving this end. It is our intention to begin this development in the near future and provide empirical backup to the theoretical background of the LFG-DOT systems presented in this thesis.
Chapter 7

Conclusions

This thesis presents a number of new hybrid models of translation based on LFG-DOP. The first, LFG-DOT1, uses LFG-DOP for the source and target language models, but imports the $\tau$-equations from LFG-MT as the translation relation. LFG-DOT1 improves on DOT1, which is not guaranteed to produce the correct translation when this is non-compositional and considerably less probable than the default, compositional alternative. DOT1’s adherence to left-most substitution in the target given a priori left-most substitution in the source is too strictly linked to the linear order of words. As soon as this deviates to any significant degree between languages, DOT1 has a significant bias in favour of the incorrect translation.

LFG-DOT1 improves the robustness of LFG-MT through the use of the LFG-DOP Discard operator, which produces generalized fragments by discarding certain f-structure features. It can, therefore, deal with ill-formed or previously unseen input where LFG-MT cannot. Unsurprisingly, however, all of the other problems of LFG-MT are maintained in LFG-DOT1. We described in chapter 3 how the correspondence-based approach faces problems when confronted with a range of translation phenomena, especially headswitching data. LFG-DOT1 fails to cope with these problems in exactly the same way as LFG-MT.

DOT2 addresses the failings of DOT1 by redefining the composition operation and providing an improved probabilistic model. It appears that in contrast to DOT1, DOT2 cannot fail to produce correct candidate translations, along with some possible wrong alternatives, depending of course on the corpus from which fragments are derived. Given this, we augmented LFG-DOT1 with the $\gamma$ function from DOT2 to give an improved model of translation. LFG-DOT2 maintains the $\tau$ translation relation to increase the chances of the correct translation being produced. We hypothesized a number of ways in which the two functions might best co-operate, with the $\gamma$ relation taking priority. Ultimately, given that the $\tau$-equations fail to derive the correct translation in all cases, we omit the $\tau$ translation relation from our subsequent models.

LFG-DOT3 relies wholly on $\gamma$ to express the translation relation, and uses f-structure information purely for monolingual filtering. The presence of this functional information prevents the formation of certain ill-formed structures which can be produced in DOT2. LFG-DOT3, therefore, has a notion of grammaticality which is missing from DOT2. Importantly, this can be used to guide the probability models in the manner required. We showed that LFG-DOT3 and DOT2 are able to cope with some of the ‘hard’ translation cases
discussed in chapters 1, 3 and 5. However, both models suffer from limited compositionality, so that in some cases the minimal statement of the translation relation is impossible.

LFG-DOT4 adds an ‘Extended Transfer’ phase to LFG-DOT3 by producing lemmatized forms using a second application of Discard. This extension overcomes the problem of limited compositionality, enabling the statement of the translation relation in an intuitive, concise fashion.

Stating the translation relation solely between (source, target) trees, as in LFG-DOT3 and LFG-DOT4, works so successfully as we are freed from the restriction of having to relate local trees. We described in chapter 1 how the CAT model of translation fails on combinations of exceptional phenomena owing to its local tree restriction. In certain cases nodes need to be mentioned in translation rules despite their having nothing to do with the translation problem being dealt with: such nodes impinge on other rules designed to deal with the problematic translational phenomena in isolation.

LFG-DOT4, like the other LFG-DOT models, is a robust system. One needs, however, to ensure that the structures obtained via Discard are of use, especially on the target side. These structures subsume the structures required for the successful generation of a well-formed target string. We described how lemmatized forms occurring in target c-structures may be replaced by the appropriate surface form by comparing the partial f-structure information to which they are linked to other, complete f-structures in the treebank. Finally, we have shown that the fragmentation issue is less of a problem for DOP-based translational treebanks than for monolingual treebanks. Nevertheless, we need to ensure that the amount of fragments produced by Discard are restricted to a small, fixed part of the probability space in our models in order to ensure a preference for grammatical structures. We have proposed the adoption of the Good-Turing method to cope with unknown words, to limit the scope of Discard-generated fragments, and to limit the amount of the probability space available to lemmatized translation pairs.

As to future work, the ideas presented in this thesis need to be tested more thoroughly on large-scale LFG-DOT corpora. We described ongoing efforts in this area in section 4.7. Given the small corpora from which our findings are derived, any results must be treated with some equivocation. Nevertheless, we consider that the issues dealt with in this thesis are relevant on a number of levels: robustness and disambiguation are of concern to practitioners of NLP, providing context-sensitive stochastic language models is of interest to computational linguists, and automated solutions to parsing and disambiguation problems are sought after in the area of machine learning. Finally, it is our hope that some of the insights provided here into difficult translation problems may be of interest to researchers in the area of LFG, and that designers of machine translation systems may derive some benefit from the treatment of such problems using the methodology of LFG-DOP.
Bibliography


256


