

TransBooster: Black Box Optimisation of Machine Translation Systems

Bart Mellebeek

BA., MSc.

A dissertation submitted in partial fulfilment of the
requirements for the award of

Doctor of Philosophy

to the

DCU

Dublin City University

School of Computing

Supervisors: Prof. Andy Way
Prof. Josef van Genabith

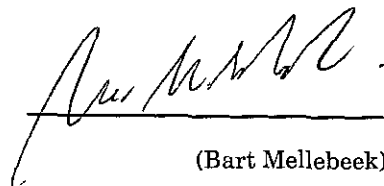
July 2007

100

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work

Signed

A handwritten signature in black ink, appearing to read 'Bart Mellebeek', written over a horizontal line.

(Bart Mellebeek)

Student ID

53144651

Date

July 2007

Contents

Abstract	vi
Acknowledgements	vii
List of Tables	viii
List of Figures	xii
1 Introduction	1
2 MT by Recursive Sentence Decomposition: Rationale	5
2.1 Introduction	5
2.2 Approaches to MT Potential and Limitations	6
2.3 TransBooster: Basics	13
2.4 Related Research	20
2.5 Summary	23
3 Methodology: Baseline MT Systems, Development Phases, Evaluation	24
3.1 Introduction	24
3.2 Baseline MT Systems	24
3.3 Development Phases	26
3.4 Evaluation	27
3.4.1 Evaluation Metrics	27
3.4.1.1 BLEU	28
3.4.1.2 NIST	30
3.4.1.3 GTM	32
3.4.1.4 Statistical Significance	34

3.4.1.5	Manual Evaluation	34
3.4.2	Experimental Set-up	35
3.5	Summary	37
4	TransBooster Architecture: Outline	39
4.1	Introduction	39
4.2	Outline	39
4.2.1	Flattening Penn-II Trees into TransBooster Trees	40
4.2.2	Finding the Pivot	42
4.2.3	Locating Satellites	43
4.2.4	Skeletons and Substitution Variables	44
4.2.5	Translating Satellites: Context	47
4.2.6	Recursion	50
4.2.7	A Worked Example	51
4.3	Substitution Variables	54
4.3.1	Introduction	54
4.3.2	Early vs. Late MT Access	54
4.3.3	Static vs. Dynamic Substitution Variables	55
4.3.4	Effects of SSV Schemata on Translation Quality	56
4.3.4.1	SSVs	57
4.3.4.2	Experimental Setup	58
4.3.4.3	Results	62
4.3.4.4	Analysis	64
4.3.5	Conclusion	66
4.4	Summary	66
5	TransBooster Architecture: Technical Details	68
5.1	Introduction	68
5.2	TransBooster Mark I	68
5.2.1	Identifying Heads	69
5.2.2	Constructing Pivots	71

5.2.2.1	Constructing Pivots. Default Tree Flattening	73
5.2.2.2	Constructing pivots: Extended Tree Flattening	74
5.2.3	Arguments vs. Adjuncts	76
5.2.4	Substitution Variables: Static vs Dynamic	78
5.2.4.1	Identifying Optimal Substitution Variables	79
5.2.5	Context: Static vs Dynamic	81
5.2.6	Chunks and their Translation	85
5.2.7	Safety Measures	90
5.2.8	Algorithm	91
5.2.8.1	Worked Example	91
5.3	TransBooster Mark II	95
5.3.1	Mark I vs Mark II	96
5.3.2	Algorithm	98
5.4	Summary	100
6	Experimental Results and Analysis	101
6.1	Introduction	101
6.2	Results for Rule-based MT	101
6.2.1	Experimental setup	101
6.2.2	Experiments with TransBooster Mark I	102
6.2.2.1	Automatic evaluation	103
6.2.2.2	Manual Evaluation	105
6.2.2.3	Analysis	109
6.2.2.4	The impact of parser-based input	111
6.2.3	Experiments with TransBooster Mark II	116
6.2.4	TransBooster and Rule-based MT: conclusion	117
6.3	Results for Data-driven MT	117
6.3.1	TransBooster and SMT	118
6.3.1.1	Experimental setup	118
6.3.1.2	Results	120
6.3.1.3	Analysis	122

6.3.2	TransBooster and EBMT	124
6.3.2.1	Marker-based EBMT	124
6.3.2.2	Experimental setup	125
6.3.2.3	Results	126
6.3.2.4	Analysis	127
6.4	Summary	128
7	TransBooster as an MEMT interface	130
7.1	Introduction	130
7.2	Multi-engine Machine Translation	131
7.2.1	Introduction	131
7.2.2	Previous Approaches to MEMT	132
7.3	TransBooster as an MEMT interface	135
7.3.1	Algorithm. Overview	135
7.3.2	Algorithm: Details	136
7.3.3	A Worked Example	139
7.4	Experimental Results and Analysis	143
7.4.1	Experimental Setup	143
7.4.2	Results	144
7.4.2.1	Human parse-annotated input	145
7.4.2.2	Input parsed by (Charniak, 2000)	147
7.4.2.3	Input parsed by (Bikel, 2002)	149
7.5	Summary	150
8	Conclusions	151
8.1	Future Work	153
	Appendices	154
A	Tags and Phrase Labels in the Penn-II Treebank	155
B	Extended Pivot Selection per Category	157

C ARG/ADJ distinction heuristics	160
D Static Substitution Variables per Category	164
E Static Context Templates per Category	171
F Implementation: Class Diagram	177
Bibliography	180

Abstract

Machine Translation (MT) systems tend to underperform when faced with long, linguistically complex sentences. Rule-based systems often trade a broad but shallow linguistic coverage for a deep, fine-grained analysis since hand-crafting rules based on detailed linguistic analyses is time-consuming, error-prone and expensive. Most data-driven systems lack the necessary syntactic knowledge to effectively deal with non-local grammatical phenomena. Therefore, both rule-based and data-driven MT systems are better at handling short, simple sentences than linguistically complex ones

This thesis proposes a new and modular approach to help MT systems improve their output quality by reducing the number of complexities in the input. Instead of trying to reinvent the wheel by proposing yet another approach to MT, we build on the strengths of existing MT paradigms while trying to remedy their shortcomings as much as possible. We do this by developing TransBooster, a wrapper technology that reduces the complexity of the MT input by a recursive decomposition algorithm which produces simple input chunks that are spoon-fed to a baseline MT system. TransBooster is not an MT system itself: it does not perform automatic translation, but operates on top of an existing MT system, guiding it through the input and trying to help the baseline system to improve the quality of its own translations through automatic complexity reduction.

In this dissertation, we outline the motivation behind TransBooster, explain its development in depth and investigate its impact on the three most important paradigms in the field: Rule-based, Example-based and Statistical MT. In addition, we use the TransBooster architecture as a promising alternative to current Multi-Engine MT techniques. We evaluate TransBooster on the language pair English→Spanish with a combination of automatic and manual evaluation metrics, providing a rigorous analysis of the potential and shortcomings of our approach.

Acknowledgements

There are many people who helped me with this thesis during the past four years. First and foremost, my supervisors Josef van Genabith and Andy Way. Without Josef, TransBooster wouldn't exist. He was the first person to bring up the possibility of recursive complexity reduction for Machine Translation and with his sharp ideas and guidance, he has been an excellent mentor throughout the course of this Ph.D. Andy's expertise, his constant encouragement and reassurance have been extremely important in helping me to keep on going. Thanks again for guiding me through those final months, Andy.

I am grateful to my examiners Mary Hearne and Paul Bennett, whose feedback has helped me to deliver a better dissertation.

Parts of this research was presented at various conferences: EAMT05, MT Summit X, IWCL06, EAMT06 and AMTA06. Thanks to the reviewers for their insightful comments that led to improvements.

Thanks also to the staff and postgraduate students at the National Centre for Language Technology and the School of Computing at DCU. Special thanks to Grzegorz, Djamé and Nicolas for their friendship, support and for the good times.

Finally, I would like to mention the names of the people who participated in the manual evaluation of the experimental results: Eva Martínez, Grzegorz (again), Roser Morante, Marta Carulla, Mireia Bartels, Marina Sánchez, Rocío del Pozo and Laura Calvelo.

Thanks to all of you.

List of Tables

3.1	Extract from the instructions for the translation of the test set.	36
4.1	Substitution Variables for NP-type constituents	57
4.2	Subcategorisable syntactic functions in LFG.	59
4.3	The 10 most frequent verbal subcategorisation frames in the Penn Treebank, in descending frequency and excluding subcategorisation frames containing only subjects	59
4.4	A test set containing a reference sentence and 5 test sentences for a particular frame-lemma pair. l = number of arguments to left of pivot, r = number of arguments to right of pivot, $1 \leq i \leq 5$	60
4.5	A test set containing a reference sentence and 5 test sentences for the frame-lemma pair <i>include</i> (<i>{subj,obj}</i>)	60
4.6	Counts for the 10 most frequent subcategorisation frames	61
4.7	Results of SSV replacement on translation quality for LogoMedia	63
4.8	Results of SSV replacement on translation quality for Systran	63
4.9	Results of SSV replacement on translation quality for SDL	64
4.10	Results of SSV replacement on translation quality for PromT	64
4.11	Translation of the test set for the frame-lemma pair <i>strap</i> (<i>{obj,subj}</i>) by Logomedia	65
4.12	Translation of the test set for the frame-lemma pair <i>face</i> (<i>{obj,subj}</i>) by Logomedia	65
5.1	Tree Head Table – the list of head-finding rules based on (Magerman, 1995)	70
5.2	Some examples of satellite chunks and their DSVs.	80
5.3	Chunks in module <i>Chunk</i> and their default context retrieval	86
6.1	TransBooster program parameters, their definition and the pages in the thesis where they are explained.	102

6.2	Optimal parameter settings per-baseline MT system.	103
6.3	TransBooster results on the 800-sentence test set with optimal parameters	103
6.4	TransBooster results on the three 600-sentence test sets with optimal parameters .	104
6.5	Impact of parameter <code>p_PivotCheck</code> on the results in Table 6.3	105
6.6	Proportion of sentences per MT engine (in the optimal setting) in which the back-off procedure is invoked at the root node. Invoking back-off at the root will disable decomposition for the entire sentence, so that the entire input is translated as is by the baseline MT system	106
6.7	Percentages of different words between TransBooster and the baseline systems on the 800-sentence test set. Figures are provided for the entire test set and for those sentences for which the back-off procedure was invoked. P is explained in Formula 6.1.	106
6.8	Number of TransBooster output sentences that are different from the baseline MT system's output.	107
6.9	Comparative results of the manual evaluation of TransBooster vs. LogoMedia, Sýstran and SDL on 200 different output sentences. B = better, S = similar, W = worse.	108
6.10	Extrapolation of the manual evaluation results in Table 6.9 for the entire 800-sentence test set. B = better, S = similar, W = worse.	108
6.11	Examples of each of the four areas of TransBooster improvements: lexical selection, word order, agreement, homograph resolution	111
6.12	Examples of sentences in which a correct complexity reduction leads to worse translation.	112
6.13	TransBooster results on 800-sentence test set, parsed with (Charniak, 2000)	113
6.14	TransBooster results on 800-sentence test set, parsed with (Bikel, 2002)	113
6.15	TransBooster results on the three 600-sentence test sets, parsed with (Charniak, 2000),	115
6.16	TransBooster results on the three 600-sentence test sets, parsed with (Bikel, 2002) .	115
6.17	TransBooster Mark II results on the 800-sentence test set.	116
6.18	Optimal parameter settings for the TransBooster-Pharaoh interface.	119

6.19	TransBooster vs Pharaoh: Results on the 800-sentence test set of Europarl	120
6.20	TransBooster vs. Pharaoh: Results on the 800-sentence test set of the WSJ	120
6.21	Comparative results of the manual evaluation of TransBooster vs Pharaoh B = better, S = similar, W = worse.	122
6.22	Extrapolation of the manual evaluation results in Table 6 21 for the entire 800- sentence test set. B = better, S = similar, W = worse.	122
6 23	Examples of improvements over Pharaoh: word order and lexical selection.	123
6 24	TransBooster vs EBMT. Results on the 800-sentence test set of Europarl	127
6.25	TransBooster vs. EBMT: Results on the 800-sentence test set of the WSJ	127
6 26	Examples of improvements over the EBMT baseline. word order and lexical selection.	128
7.1	Example sentence (20). result of TB _{MEI} vs. baseline MT engines.	142
7.2	Result of TB _{MEI} vs. baseline MT engines on the example sentence 'Imperial Corp., based in San Diego, is the parent of Imperial Savings & Loan.'	142
7 3	Result of TB _{MEI} vs baseline MT engines on the example sentence 'Mr. Pierce said Elcotel should realize a minimum of \$10 of recurring net earnings for each machine each month.'	143
7.4	Results of the three baseline MT systems on the 800-sentence test set: absolute scores (cf Table 6 3 in Chapter 6) on page 101)	145
7 5	TB _{MEI} vs TB _{SEI} absolute scores for human parse-annotated input	146
7.6	TB _{MEI} vs. TB _{SEI} and baseline systems: relative scores for human parse-annotated input	146
7.7	TB _{SEI} vs baseline systems relative scores for human parse-annotated input . . .	147
7.8	Relative contribution of each of the selection heuristics for the results in Table 7.5. .	147
7.9	TB _{MEI} and TB _{SEI} absolute scores for input parsed by (Charniak, 2000)	148
7.10	TB _{MEI} vs TB _{SEI} and baseline systems: relative scores for input parsed by (Charniak, 2000)	148
7.11	Relative contribution of each of the selection heuristics for the results in Table 7.9 .	148
7.12	TB _{MEI} and TB _{SEI} absolute scores for input parsed by (Bikel, 2002)	149
7.13	TB _{MEI} vs TB _{SEI} and baseline systems relative scores for input parsed by (Bikel, 2002)	149

7.14	Relative contribution of each of the selection heuristics for the results in Table 7.12.	150
A.1	Tag labels in the Penn-II Treebank	156
A.2	Phrase labels in the Penn-II Treebank.	156
B.1	Nr of rule types (covering 85% of rule tokens) and basic extended pivot treatment for non-terminal nodes in the Penn-II Treebank. Parentheses indicate optional categories.	159
C.1	ARG/ADJ distinction heuristics per category, independent of the mother node. . .	161
C.2	ARG/ADJ distinction heuristics per category, dependent of the mother node. . .	163
D.1	Static Substitution Variables per Category.	170
E.1	Static Context Templates per Category.	176
F.1	Language-dependent vs. Language-independent Elements in TransBooster.	179

List of Figures

2.1	The Vauquois MT triangle	7
2.2	Wu's 3-D model space for MT.	11
2.3	Trajectory of historical development of RBMT, SMT and EBMT systems, respectively represented by triangles, dots and squares, according to (Wu, 2005).	12
2.4	TransBooster interfacing with baseline MT system.	13
2.5	TransBooster in Wu's 3-D model space for MT The arrows represent the fact that TransBooster can be interfaced with all types of baseline MT systems.	20
3.1	Bitext grid illustrating the relationship between an example candidate translation and its corresponding reference translation. Each bullet or 'hit' indicates a word contained in both the candidate and reference texts.	32
3.2	Bitext representing two different candidate texts for the same reference text The MMS in Equation 3.7 rewards the better word order in candidate text (b) by weighting each contiguous sequence of matching words by their length, which is indicated by the greater surface of shaded area in (b).	34
3.3	A section of the web page for translators to construct the gold standard reference translations.	37
4.1	Flattening a Penn-II tree into a TransBooster tree. l = number of satellites to left of pivot. r = number of satellites to right of pivot.	41
4.2	Penn-II tree representation of 'The chairman, a long-time rival of Bill Gates, likes fast and confidential deals'	41
4.3	Flattened TransBooster tree obtained from Penn-II structure in Figure 4.2	41
4.4	Penn-II tree representation of 'might have to buy a large quantity of sugar'	42
4.5	Penn-II tree representation of 'close to the utility industry.'	43

4.6	The recursive nature of the TransBooster decomposition. each satellite chunk SAT_i is decomposed until only optimal chunks remain.	51
4.7	Penn-II tree representation of 'Imperial Corp, based in San Diego, is the parent of Imperial Savings & Loan.'	52
4.8	TransBooster tree representation of (4.7).	52
5.1	Basic tree flattening 1-7 are arbitrary non-terminal categories A-L are lexical items. Node 3 is the head of node 1. Node 4 is the head of node 3. The resulting flattened tree on the right-hand side is the input to TransBooster's decomposition module.	74
5.2	Extended tree flattening 1-7 are arbitrary non-terminal categories A-L are lexical items. Node 3 is the head of node 1. Node 4 is the head of node 3.	75
5.3	Penn-II tree representation of 'we were coming down into their canal.'	76
5.4	Penn-II tree representation of 'Individual investors have turned away from the stock market'	77
5.5	Parse tree representation of 'The chairman, a long-time rival of Bill Gates, likes fast and confidential deals'	82
5.6	Parse tree representation of node $S-TPC-1$ in (61)	83
5.7	The back-end of the TransBooster Engine.	85
5.8	The (in theory) never-ending cycle of dynamic context template translations	88
5.9	The three stages in a TransBooster run.	89
5.10	The standard TransBooster algorithm (TB_{MarkI}) in pseudo-code	91
5.11	TransBooster tree representation of (67)	92
5.12	Input Chunk S into decomposition algorithm of TB_{MarkII}	96
5.13	The simplified TransBooster algorithm (TB_{MarkII}) in pseudo-code.	99
6.1	The human parse-annotated structure of the chunk 'a Belgian pretending to be Italian' in the Penn-II Treebank.	114
6.2	The parser output of (Bikel, 2002) of the chunk 'a Belgian pretending to be Italian'	114
7.1	An example English sentence and its translation from five different MT systems, from (Bangalore et al, 2001)	133

7.2	Lattice representation of the example sentence in Figure 7 1, from (Bangalore et al , 2001)	134
7.3	A flow chart of the entire MEMT system, with C_i the i^{th} input chunk ($1 \leq i \leq M$), E_j the j^{th} MT engine ($1 \leq j \leq N$) and $C_{i,j}$ the translation of C_i by E_j	136
7.4	Output of example sentence (20) by the three baseline MT engines: LogoMedia, Systran and SDL	139
7.5	Decomposition of Input.	141
7.6	Selection of best output chunk. The optimal combination follows the arcs in bold.	141
F.1	Implementation of TransBooster Application (Java version J2SE 5.0): class diagram.	177

Chapter 1

Introduction

Machine Translation (MT) has been an active area of research in Artificial Intelligence (AI) since the 1950s. Over the years, initial overinflated expectations (*'the production of fully automatic high-quality translations in an unrestricted domain'*) have been scaled down due to the complexity of modeling the human translation process. In recent years, the quality achieved by MT systems is sufficient to make MT commercially viable, not as a substitute for the human translator, but as a possibly useful time-saving component of a translation process that involves other important components (such as translation memories, on-line dictionaries, terminology management systems and human post-editing).

Most of the existing commercial MT systems are implemented based on the rule-based transfer paradigm (RBMT). The main theoretical limitation of this paradigm is that transfer rules alone are not sufficient to replace the real-world knowledge that humans use to perform translation (Bar-Hillel, 1960). In addition, hand-crafting rules based on detailed linguistic analyses is time-consuming, error-prone and expensive. Therefore, commercial RBMT systems tend to trade a broad but shallow linguistic coverage for a deep, fine-grained analysis. As a consequence, most existing commercial MT systems do not perform to the best of their abilities: they are more successful in translating short, simple sentences than long and complex ones. The longer the input sentence, the more likely the MT system will be led astray by the lexical, syntactic and semantic complexities in the source and target languages. When MT systems fail to produce a complete analysis of the input, their recovery strategies for rendering a translation often result in 'word

salad’

In this dissertation, we investigate whether it is possible to help MT systems improve their translations by reducing the number of complexities in the input. Instead of trying to reinvent the wheel by proposing yet another approach to MT, we build on the strengths of existing MT paradigms while trying to remedy their shortcomings as much as possible. We do this by developing TransBooster, a wrapper technology that reduces the complexity of the MT input by a recursive decomposition process which produces simple input chunks that are spoon-fed to a baseline MT system. In other words, the objective of TransBooster is to enhance the quality of existing current MT technology through a divide-and-conquer approach. We verify whether the reduction in complexity provided by TransBooster is sufficient to achieve this goal.

This thesis is *not* about the development of an MT system. It describes the theory behind and the deployment of a wrapper technology to be used *on top of* existing MT systems. This is a new area of research in MT with little related previous publications. Thurmair (1992) and Gerber and Hovy (1998) experimented with similar ideas, as we will explain in Chapter 2, but to the best of our knowledge, this is the first large-scale attempt to improve MT output through automatic complexity reduction.

In order to test the possible advantages of recursive sentence decomposition for a particular MT system, it would have been possible to design an application for that particular MT system by using the knowledge of its internal workings. Instead, we chose to treat the MT systems that were interfaced to TransBooster as *‘black boxes’*. This has the advantage that the TransBooster technology can be used on top of all sorts of different MT systems, regardless of their implementation or of the MT paradigm that they adhere to.

During the development of TransBooster, human parse-annotated sentences of the Penn-II Treebank (Marcus et al., 1994) were used as input. The results obtained on this ‘perfectly annotated’ input constitute a theoretical upper bound for the improvements that are possible for unannotated text, which has to be parsed automatically as a necessary step prior to decomposition. Once the TransBooster algorithm was finalised, we performed experiments with the output of two state-of-the-art statistical parsers ((Charniak, 2000) and (Bikel, 2002)). Current state-of-the-art probabilistic parsing technology is capable of

providing tree-based precision & recall scores of around 90% and dependency-based scores of around 80%. The experiments conducted will show whether the possible advantages through complexity reduction outweigh the inevitable errors and noise introduced by even the best available parsers.

Although the majority of the commercially available MT systems are (still) rule-based, most of the current research in MT is corpus-based, with Statistical Machine Translation (SMT) and Example-Based Machine Translation (EBMT) being the predominant research paradigms. Since most of the currently available data-driven systems are not able to efficiently deal with non-local syntactic phenomena, long and syntactically complex sentences pose a significant challenge to both SMT and EBMT. Therefore, after experimenting with TransBooster on top of three RBMT systems, we investigate the effects of TransBooster's complexity reduction on a phrase-based SMT system and a marker-based EBMT system.

In addition, given that TransBooster is independent of the internal workings of its client MT systems, it is possible to interface it simultaneously with several MT engines. In Chapter 7 we explain how we adapted TransBooster as a Multi-Engine Machine Translation (MEMT) interface and analyse its performance.

This thesis is structured as follows:

Chapter 2 explains the rationale of recursive sentence decomposition for MT and compares the TransBooster approach to other MT paradigms.

Chapter 3 introduces the baseline MT systems used throughout this dissertation and explains how the performance of TransBooster is measured.

Chapter 4 contains a general outline of the TransBooster architecture.

Chapter 5 describes in depth the concepts introduced in Chapter 4 and explains the technical details of two different TransBooster implementations.

Chapter 6 analyses the experimental results of the TransBooster output in comparison with the baseline MT systems that were introduced in Chapter 3.

Chapter 7 examines the use of TransBooster as an MEMT interface.

Chapter 8 concludes and outlines possible areas of future research.

The research presented in this dissertation was published in several peer-reviewed Conference Proceedings. (Mellebeek et al , 2005a) and (Mellebeek et al , 2005b) present the basics of the TransBooster architecture and show its performance with respect to rule-based MT. Subsequently, TransBooster was adapted for integration with data-driven MT systems, the results of which are published in (Mellebeek et al , 2006a) for SMT and (Owczarzak et al., 2006)¹ for EBMT. (Armstrong et al , 2006) contains more information on how the baseline system for the EBMT experiments was constructed. Finally, (Mellebeek et al., 2006b) analyses the use of TransBooster as an MEMT interface.

¹Although most of the experimental work for this paper was carried out by my colleague K Owczarzak, the background algorithms and design are largely my own work.

Chapter 2

MT by Recursive Sentence Decomposition: Rationale

2.1 Introduction

Fully Automatic High-Quality Machine Translation (FAHQMT) in an unrestricted domain is considered an AI-complete problem¹ by many researchers (Trujillo, 1999), since solving this problem seems to require the equivalent of human intelligence. Instead of pursuing the futile quest for this ‘holy grail’, contemporary research in Machine Translation (MT) focuses on trying to make MT useful rather than perfect.

If we are allowed to omit one of the three above-mentioned requirements (‘fully-automatic’, ‘high quality’, ‘unrestricted domain’), then it is uncontroversial to assert that useful MT systems have already been achieved

1. FAHQMT systems have been developed for *restricted* domains such as weather reports (Chandioux, 1976; Chandioux and Grimaila, 1996) and heavy equipment manuals (Nyberg and Mitamura, 1992) amongst others.
2. High-quality MT in unrestricted domains is feasible if the MT system is ‘aided’ by human post-editing (Krings, 2001). Also, the use of controlled language (Bernth and Gdanec, 2001; O’Brien, 2003), designed to eliminate the maximum amount of ambiguities in the input, can lead to a significantly improved MT output. The use

¹A problem is defined as AI-complete if its solution requires a solution to every major problem in AI.

of controlled language-for internal documentation is common practice nowadays in many major companies (e.g. SIEMENS (Lehrndorfer and Schachtl, 1998) or FORD (Rychtycky, 2002) to mention only a few).

3. Fully-automatic MT in an unrestricted domain rarely produces high quality output, as one can easily verify when using one of the many on-line MT engines that populate the World Wide Web. Nevertheless, a less-than-perfect translation can be sufficient for a user interested in the overall gist of the contents of the source text

TransBooster is situated in this third domain: it was initially designed to improve the output of fully-automatic wide-coverage MT systems. In this chapter, we motivate the rationale behind TransBooster. Section 2.2 contains a brief analysis of the potential and limitations of the most important approaches to MT. In Section 2.3, we explain how TransBooster can help MT systems improve their *own* output by reducing the complexity of the input. We also situate our approach with respect to the other approaches to MT in the three-dimensional MT model space of (Wu, 2005). Finally, in Section 2.4, we analyse the similarities/differences of TransBooster with previously published related research.

2.2 Approaches to MT: Potential and Limitations

Approaches to MT are usually categorised as either rule-based (RBMT) or corpus-based (CBMT): RBMT systems employ rules hand-crafted by humans to perform translation, whereas CBMT systems use machine learning techniques to induce translation knowledge from bilingual aligned corpora. Up until the mid 1980s, the vast majority of MT research/production was rule-based. The following three RBMT approaches are commonly distinguished², depending on the degree of abstraction of their intermediate representation in the well-known Vaquouis MT triangle, represented in Figure 2.1.

Direct RBMT Direct MT systems lack any kind of intermediate stages in the translation process. After a limited morphological analysis of the source language (SL) sentence,

²We abstract away from the fact that for each of the three approaches, the 'rules' could be automatically induced from corpora instead of hand-coded by experts (e.g. (Menezes and Richardson, 2001; Xia and McCord, 2004)), which would place them effectively in the CBMT paradigm. In the rest of this work, we will interpret these approaches as RBMT.

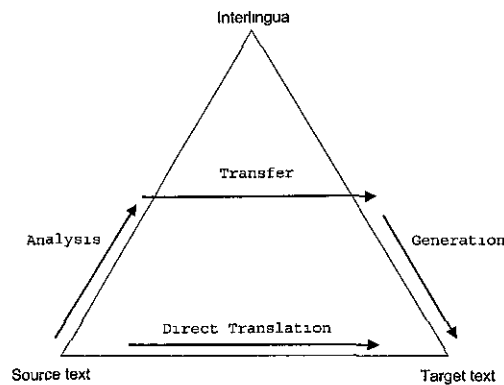


Figure 2.1: The Vauquois MT triangle.

a translation for each SL word is selected in a bilingual dictionary and a certain local reordering in target might take place. Many of the early MT systems were based on this approach.

POTENTIAL:

- relatively easy to implement.

LIMITATIONS:

- severe limitations in ambiguity resolution and correct word order generation.
- limited scalability: for n languages, $n(n - 1)$ different entire systems have to be implemented.

Transfer-based RBMT Transfer-based MT systems relate source and target language (TL) at the level of syntax. An analysis module produces the intermediate syntactic representation, a transfer module finds a corresponding syntactic structure in the TL and a generation module generates a TL output. Transfer-based MT was the most popular research paradigm until the late 1980s. Most of the currently available commercial MT systems were designed based on this approach (e.g. METAL (Bennett and Slocum, 1985) or Systran (Senellart et al., 2001) to mention only a few).

POTENTIAL:

- improved ambiguity resolution and treatment of syntactic phenomena, especially for closely related languages

- improved scalability with respect to Direct RBMT: for n languages, n analysis and generation modules and $n(n-1)$ transfer modules have to be implemented.

LIMITATIONS:

- lack of coverage: grammars designed on crude heuristics and tested on toy sentences, limited lexica.
- coverage expansion is problematic and could lead to an increase in ambiguity, since new rules might interfere with old ones. Danger of over-analysis and over-generation.
- a huge amount of rules and extensive lexica are time-consuming to build and error-prone.
- the task is much more difficult and less successful where the intermediate structural representations differ to any great degree between SL and TL.

Interlingua-based RBMT Interlingua systems try to incorporate a universal meaning representation which renders a language-dependent transfer phase unnecessary, using only an analysis phase into and a generation phase from this language-independent interlingua to produce correct translations. The problem of finding an adequate meaning representation for all languages is closely related to knowledge-representation problems in classical AI, one of the major difficulties in the field. Research in interlingua MT was popular during the 1980s and early 1990s (e.g. ROSETTA (Landsbergen, 1989), KBMT (Goodman and Nirenburg, 1991), PANGLOSS (Frederking et al., 1993)), but has now largely been abandoned due to the complexity of the task.

POTENTIAL:

- in theory, *the* solution to MT, since real-world knowledge appears to be a prerequisite for FAHQMT (Bar-Hillel, 1960).
- perfect scalability for n languages, n analysis and generation modules have to be implemented.

LIMITATIONS:

- not feasible in practice due to the difficulty of the knowledge representation problem.

During the 1980s, a number of factors led to a resurgence of interest in empirical techniques in Natural Language Processing (NLP): (i) the symbolic, rule-based approach had proven insufficient to provide high-quality solutions for most NLP problems, (ii) the emergence of computers with sufficient speed and memory to handle large amounts of data, and (iii) the availability of large-scale machine-readable corpora.

In MT, the two exponents of the empirical or corpus-based approach are Statistical MT (SMT) and Example-Based MT (EBMT).

Statistical MT (SMT) SMT uses bilingual aligned corpora and probability models of translation to estimate the most likely language output sentence e , given a foreign input sentence f , or in other words $\operatorname{argmax}P(e|f)$. In the early word-based IBM models (Brown et al., 1993), this probability was estimated by decomposing $\operatorname{argmax}P(e|f)$, as an instance of the noisy-channel approach, into two sources of information, a translation model $P(f|e)$ and a language model $P(e)$:

$$\operatorname{argmax}P(e|f) = \operatorname{argmax}P(e)P(f|e) \quad (2.1)$$

Later SMT research improved the performance of the early SMT attempts by incorporating *phrases* or sequences of words into the models in a variety of ways ((Yamada and Knight, 2001; Och and Ney, 2002; Marcu and Wong, 2002; Koehn et al., 2003; Chiang, 2005) to mention only a few). At this moment, SMT is the dominant research area in Machine Translation.

POTENTIAL:

- minimal human effort, easy to build.
- elegant way to deal with idioms and local ambiguities, both lexical and structural.
- robust.

LIMITATIONS:

- domain specificity: highly dependent on training corpus.
- the need for large amounts of training data.
- syntactically limited: only local syntactic phenomena can be dealt with.

Example-based MT (EBMT) Machine Translation by analogy. The SL sentence is split up into a number of chunks, which are matched against fragments in a bilingual aligned corpus. After identifying the corresponding translation fragments in target, these are recombined into the appropriate target text. EBMT started with (Nagao, 1984) but research did not take off until the late 1980s. Nowadays, it is the second most important research paradigm in the field, after SMT. Cf. (Carl and Way, 2003) for an overview of recent approaches in EBMT.

POTENTIAL AND LIMITATIONS

- similar to SMT limitations. EBMT systems generally do not require as much training data as SMT systems but rely on deeper linguistic information, which might not be trivial to extract.

Few MT systems are 'pure' implementations of one of the above-mentioned approaches. Often, they combine several techniques into a hybrid solution. For example, it is common for modern RBMT systems to incorporate a certain amount of statistical techniques for word-sense disambiguation or the translation of idioms. Certain types of SMT systems try to incorporate linguistic knowledge in their models (Yamada and Knight, 2001; Charniak et al., 2003; Burbank et al., 2005; Chiang, 2005). Also, it is not uncommon for EBMT systems to use techniques specific to the SMT community (Groves and Way, 2005; Menezes and Quirk, 2005; Armstrong et al., 2006).

Heated debates over whether a particular system is SMT, EBMT or RBMT at its core are not uncommon in the MT community. Such discussions are sometimes motivated by rather subjective criteria and do not contribute to a better understanding of the similarities and differences of the various MT approaches. Wu (2005) provides an elegant, formal solution for this problem: He suggests a three-dimensional *MT model space* in which to situate MT approaches so that their differences acquire a clear graphical dimension. The three axes in the *MT model space*, as represented in Figure 2.2, correspond to the

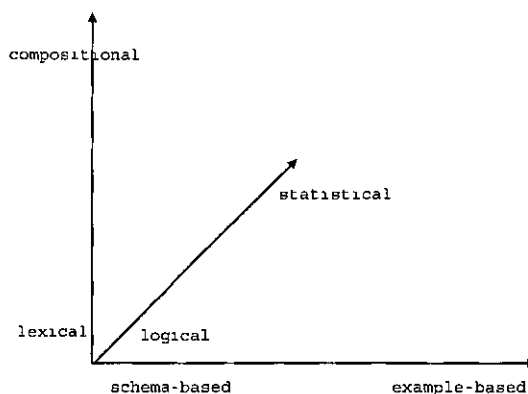


Figure 2.2: Wu's 3-D model space for MT.

formal dichotomies of (i) compositional vs. lexical, (ii) statistical vs. logical, and (iii) example-based vs. schema-based

The COMPOSITIONAL VS. LEXICAL axis measures the level of compositionality in the bilingual *transfer rules* of an MT system: compositional transfer rules declaratively describe how larger chunks can be translated by recursively composing smaller translated chunks, whereas lexical transfer rules directly translate lexical items into their target equivalents. The STATISTICAL VS. LOGICAL axis represents the extent to which mathematical statistics and probability are used in the MT system. The EXAMPLE-BASED VS. SCHEMA-BASED axis indicates whether translation is performed based on a large library of examples or on abstract schemata.

Wu (2005) plots the trajectory of the historical development of a number of MT approaches in the 3-D model space, an adaptation of which is presented in Figure 2.3. In this figure, RBMT systems are represented by triangles, SMT systems by circles and EBMT systems by squares. The evolution in RBMT systems moves from highly compositional and logical systems (Locke and Booth, 1955) to slightly more lexical systems (Chandioux, 1976; Maas, 1987), incorporating more statistics along the way (Senellart et al., 2001). SMT systems move from the word-based IBM models (Brown et al., 1993) towards more compositional and example-based models (Wu and Wong, 1998; Yamada and Knight, 2001; Och and Ney, 2002). EMBT systems evolve from pure analogy-based systems (Nagao, 1984, Lepage, 2005) to more lexical template-driven systems, with certain approaches

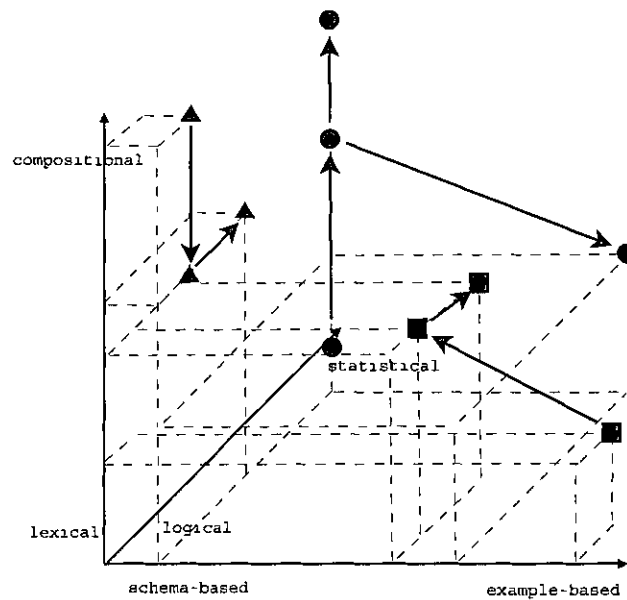


Figure 2.3: Trajectory of historical development of RBMT, SMT and EBMT systems, respectively represented by triangles, dots and squares, according to (Wu, 2005).

incorporating more statistics (Groves and Way, 2005, Menezes and Quirk, 2005).³ At the end of Section 2.3, we indicate where TransBooster is situated in this model space

The basic idea of TransBooster emerged after analysing common flaws of fully-automatic wide-coverage MT systems, such as the many on-line MT systems that populate the World Wide Web, most of which are rule-based. Since a detailed linguistic analysis of translation input is potentially costly, both in terms of development and processing time, and because of the importance of robustness for commercial MT, wide-coverage MT systems tend to trade a broad but shallow linguistic coverage for a deep, fine-grained analysis. As a consequence, most existing commercial MT systems are more successful in translating short, simple sentences than long and complex ones. The longer the input sentence, the more likely the MT system will be led astray by the lexical, syntactic and semantic complexities in the source and target languages.

If a method can be found to reduce the number of complexities in an input sentence before sending the input to an MT system, the same MT system should be able to improve

³Cf. (Wu, 2005) for the full details

the quality of its output since a reduction in complexity, in theory at any rate, relieves some of the burden on its analysis, transfer and generation modules, which are often limited to analysing local phenomena. In this thesis, we present the design, development and deployment of an application that achieves this desired complexity reduction by recursive sentence decomposition. TransBooster breaks down input sentences into smaller, syntactically simpler chunks and embeds these chunks in short context templates that mimic the context of the original sentence. TransBooster then spoon-feeds the resulting chunks to the MT system, one by one, and uses their translation to compose an output sentence.

2.3 TransBooster: Basics

TransBooster acts as a wrapper technology application: it operates on top of an existing ‘baseline’ MT system, guiding its translation, as is shown in Figure 2.4. TransBooster splits an input sentence S into N chunks $C_1 \dots C_N$, sends these chunks for translation to the baseline MT system and forms the output S' by recomposing the recovered translations $C'_1 \dots C'_N$.

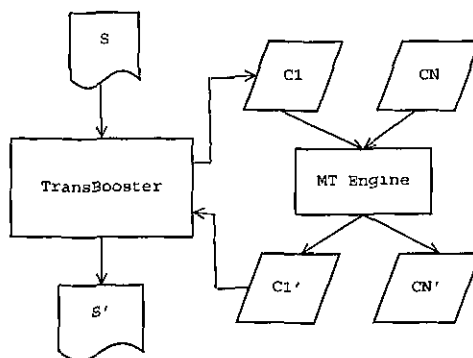


Figure 2.4: TransBooster interfacing with baseline MT system

Throughout the entire process, the baseline MT system is treated as a black box and does all the translation itself. In other words, TransBooster tries to enhance the MT system’s *own* possibilities through a divide-and-conquer approach by reducing the syntactic complexity of the input.

The fact that TransBooster does not presuppose any knowledge of the internal workings

of the baseline system used, makes it possible to interface the program with implementations of any of the different MT architectures outlined in Section 2.2. In Chapter 6, we present an analysis of the application of TransBooster on top of three widely-used commercial rule-based systems, as well as the results of interfacing our approach with an in-house-constructed phrase-based SMT system. In Chapter 7, we describe how TransBooster can be interfaced with multiple MT engines simultaneously in a multi-engine MT architecture.

The following examples illustrate the rationale behind TransBooster, namely that complexity reduction through sentence decomposition can lead to improved translations.

Example 1

Compare the translations (English→Spanish) by a human translator and the MT system developed by SDL International⁴ of the example sentence in (1)

(1) Source	'His stubbornness <u>has</u> , in fact, <u>created</u> problems where they didn't exist'
Human translator	'De hecho, <u>su terquedad ha creado</u> problemas donde antes no existían'
SDL	'Su terquedad <u>tiene</u> , de hecho, los problemas <u>creados</u> donde ellos no existieron.'

In this example, the fact that the auxiliary 'has' and the main verb 'created' are separated by the adverbial phrase 'in fact' causes the MT system to wrongly interpret 'has' as the main verb and 'created' as a past participle modifier, generating the erroneous translations 'tiene' and 'creados'.

Nonetheless, the MT system is able to correctly translate the shorter strings in (2). These strings contain decomposed parts of the input sentence (included in square brackets [...]), embedded in a suitable context. As will be explained in detail in Chapters 4 and 5, the presence of context templates mimicking the original context in which the component strings occurred is necessary to ensure a correct translation.

⁴<http://www.freetranslation.com>

- (2) a. '[His stubbornness] is nice.' → '[Su terquedad] es agradable.'
- b. 'The man [has created] cars' → 'El hombre [ha creado] coches'
- c. '[In fact], the man is sleeping' → '[De hecho], el hombre duerme.'
- d. 'The man has seen [problems where they didn't exist]' → 'El hombre ha visto [los problemas donde ellos no existieron].'

The recomposition of the translations of the component strings results in (3).

- (3) 'Su terquedad, de hecho, ha creado los problemas donde ellos no existieron'.

This recomposed translation is better than the original output produced for the complete input string by the same MT system in (1), since the removal of the ambiguity caused by the adverbial phrase 'in fact' helps the system to recognise 'has created' as a verbal unit, allowing its generation module to output the correct 'ha creado', just like the human translation.

Example 2

Compare the translations (English → German) by a human translator and the MT system Systran⁵ of the example sentence in (4):

- | | | |
|-----|------------------|--|
| (4) | Source | 'The chairman, a long-time rival of Bill Gates, <u>likes fast</u> and confidential deals' |
| | Human translator | 'Der Vorsitzende, ein langfristiger Rivale von Bill Gates, mag schnelle und vertrauliche Abkommen' |
| | Systran | 'Der Vorsitzende, ein langfristiger Rivale von Bill Gates, <u>Gleiche fasten</u> und vertrauliche Abkommen.' |

The problem in the output produced by Systran resides in a wrong homograph-resolution of 'likes' and 'fast' ('likes' is interpreted as a noun instead as a verb, and 'fast' receives a verbal interpretation instead of the correct nominal one). Although the MT output is in many respects similar to the human translation, the misinterpretation of only two items in the source sentence renders the result unintelligible. As in the previous example, breaking up the original string into simpler parts in (5) forces the MT system to improve its interpretation of the wrongly identified parts.

⁵<http://www.systransoft.com>

- (5) a. '[The chairman, a long-time rival of Bill Gates,] is sleeping.' → '[Der Vorsitzende, ein langfristiger Rivale von Bill Gates,] schläft '
- b. 'The man [likes] dogs ' → 'Der Mann [mag] Hunde.'
- c. 'The man sees [fast and confidential deals] ' → 'Der Mann sieht [die schnellen und vertraulichen Abkommen].'⁶

The recomposition of the component parts in (6) results in a significantly improved translation with respect to the original translation produced by the MT system in (4), due to the fact that the complexity reduction by decomposition helps the MT system analyse 'likes' as a verb and 'fast' as an adjective, leading to the improved translations of 'mag' and 'schnellen', respectively.

- (6) Der Vorsitzende, ein langfristiger Rivale von Bill Gates, mag die schnellen und vertraulichen Abkommen

It is not true that complexity reduction through sentence decomposition will automatically lead to improvements in all cases. Care must be taken to split a complex input sentence at the appropriate boundaries and to embed the decomposed chunks in a context that preserves enough similarities with the original to avoid mistranslations. In addition, even a perfect decomposition coupled with a correct context embedding will not automatically lead to improvements: if the baseline MT system does not contain alternatives for a given lexical item, an improved analysis or homograph resolution will not lead to a different translation for that item. The following examples demonstrate the need for caution when changing the original structure of the input sentence.

Example 3

Compare the translations (English→Spanish) by a human translator and Systran of the example sentence in (7):

⁶On the differences 'schnellen/schnelle' and 'vertraulichen/vertrauliche' when comparing this example to (4) German adjectives receive the weak inflection *-en* in the accusative plural case after the definite article 'die', as occurs in this example. When no article is used, as is shown the human translation of (4), they receive the strong inflection *-e*. Both constructions are correct

- | | |
|------------------|---|
| (7) Source | 'The nurses, nervous about their new job, handed the surgeon the wrong instruments.' |
| Human translator | 'Las enfermeras, nerviosas por su nuevo trabajo, dieron los instrumentos incorrectos al cirujano.' |
| Systran | 'Las enfermeras, nerviosas sobre su nuevo trabajo, dieron a cirujano los instrumentos incorrectos.' |

In this case, the output produced by Systran is a quite accurate and well-formed translation of the original, apart from a few minor details (the generation of the preposition 'sobre' instead of the correct 'por', the omission of the article 'el' which leads to the erroneous 'a cirujano' instead of the correct 'al cirujano'). A possible decomposition into smaller chunks could lead to (8):

- (8)
- a. '[The nurses] are sleeping.' → '[Las enfermeras] están durmiendo'
 - b. 'The man, [nervous about their new jobs].' → 'El hombre, [nervioso sobre sus nuevos trabajos]'
 - c. 'SUBJ⁷ [handed] OBJ2 OBJ1.' → 'SUBJ [dio] OBJ2 OBJ1.'
 - d. 'I see [the surgeon]' → 'Veo [a cirujano].'
 - e. 'I see [the wrong instruments]' → 'Veo [los instrumentos incorrectos].'

This leads to the recomposed translation in (9)

- (9) 'Las enfermeras, nervioso sobre sus nuevos trabajos, dio a cirujano los instrumentos incorrectos.'

In this case, the output of the recomposed translation is worse than the original translation of the entire input string in (7), since the subject-verb agreement between 'The nurses' and 'handed', as well as the head-modifier agreement between 'nurses' and 'nervous' is missing, leading to erroneous translations of 'dio' and 'nervioso'. The reason for this deterioration is the selection of too basic a placeholder for the substitution of the NP 'The nurses', as well as the use of a deficient context ('The man'), for the same NP.

Example 4

Compare the translations (English → German) by a human translator and the MT system

LogoMedia⁸ of the example sentence in (10).

⁷SUBJ, OBJ1 and OBJ2 are non-word string Substitution Variables, which will be further explained in Chapter 4 on page 57.

⁸<http://www.lec.com>

- | | | |
|------|------------------|--|
| (10) | Source | ‘The accused pleaded guilty to the corruption charges.’ |
| | Human translator | ‘Die Angeklagten bekannten sich zu den Korruptionsvorwürfen schuldig.’ |
| | LogoMedia | ‘Die Angeklagten bekannten sich schuldig zu den Korruptionsanklagen.’ |

As in the previous example, the output produced by the MT system is quite acceptable. The only minor errors are a slightly awkward word order and the fact that ‘corruption charges’ is translated as the correct but rather infrequent ‘Korruptionsanklagen’ instead of the more usual ‘Korruptionsvorwürfen’. Nevertheless, the MT output would achieve a high score when measured for accuracy and fluency.

A possible decomposition into smaller chunks could lead to (11):

- | | | |
|------|----|--|
| (11) | a | ‘[The accused] are sleeping.’ → ‘[Die Angeklagten] schlafen’ |
| | b. | ‘The men [pleaded].’ → ‘Die Männer [plädierten].’ |
| | c. | ‘I am [guilty to the corruption charges]’ → ‘Ich bin [zu den Korruptionsgebühren schuldig].’ |

This leads to the recomposed translation in (12):

- | | |
|------|--|
| (12) | ‘Die Angeklagten plädierten zu den Korruptionsgebühren schuldig’ |
|------|--|

The main reason why this translation is considerably worse than the original output of the entire input string in (10) is the splitting of the idiom ‘plead guilty’, leading to the erroneous ‘plädierten’, which is a literal translation of ‘pleaded’. In addition, the change of context of ‘guilty to the corruption charges’ causes the MT system to translate the compound ‘corruption charges’ into the nonsensical ‘Korruptionsgebühren’, instead of the quite correct ‘Korruptionsanklagen’. Probably, this is caused by the fact that the transfer module of LogoMedia uses semantic criteria to select output candidates for a lexical item: in the original sentence, the presence of the legal term ‘the accused’ causes the MT system to correctly select the legal alternative for ‘charges’, namely ‘Anklagen’, instead of the literal interpretation ‘Gebühren’, which creates the amusing impression that the accused are standing trial for paying charges to be entitled to corruption.

While examples 1 and 2 show that improvements are certainly possible, examples 3 and 4 demonstrate that the noise generated by the decomposition algorithm might lead to worse results in some cases. Therefore, the main challenge that we are faced with is to find an efficient way to maximise the complexity reduction through recursive sentence

decomposition, while, at the same time, trying to minimise the amount of noise produced by the algorithm.

The working of TransBooster is explained in detail in Chapters 4 and 5. The following is a brief resumé of its working

TransBooster decomposes an input sentence into optimal chunks by using a recursive algorithm that starts at the top-level node of the syntactic parse tree representing the input string and examines each node as it traverses the tree. The produced chunks are embedded in context templates which are at the same time sophisticated enough to yield a correct translation of the embedded chunks, and simple enough to send as simple an input as possible to the MT engine. While keeping track of the position of the translation of the chunks in target, TransBooster retrieves the translations of the embedded chunks produced by the baseline MT engine and recombines the output chunks to produce the final result, which we expect to be of higher quality than the automatic translation of the original, complete and complex input sentence.

We mentioned in Section 2.2 that we would situate TransBooster in the 3-D MT Model Space of (Wu, 2005). TransBooster is a *hyper*compositional, logical and schema-based approach to MT that can be interfaced with any type of MT system, using the MT system as if it were an internal dictionary, as is graphically presented in Figure 2.5. TransBooster does not, to any extent, rely on a library of examples at run-time: therefore it is graphically located at the very start of the *X*-axis. Although TransBooster itself does not use statistical models for decomposition or recombination, its input is produced by state-of-the-art statistical parsers: therefore we situate it in the middle of the *Z*-axis. Given that the compositionality of TransBooster is at the core of its workings and since it was designed to be primarily interfaced with MT systems that are already compositional in nature, we define it as *hyper*compositional and situate it at the extreme end of the *Y*-axis.

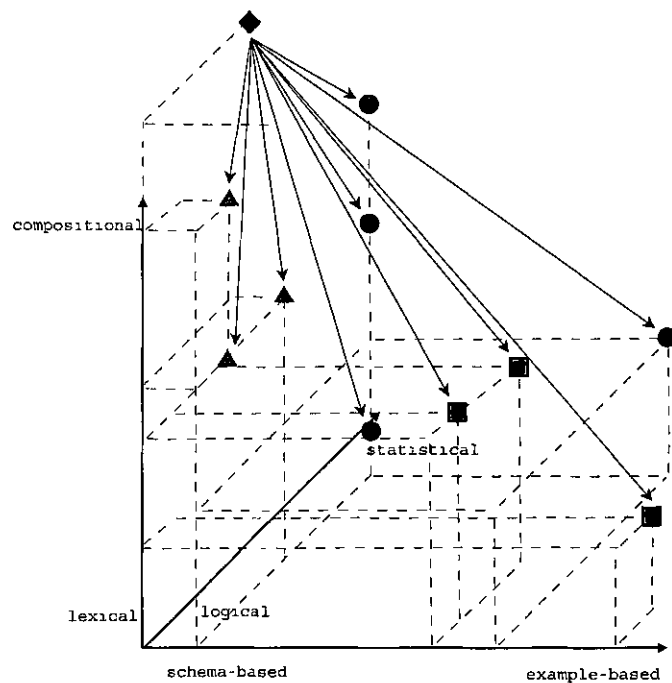


Figure 2.5 TransBooster in Wu's 3-D model space for MT. The arrows represent the fact that TransBooster can be interfaced with all types of baseline MT systems.

2.4 Related Research

During the early 1990s, research took place at the University of Leuven (Belgium) and Siemens-Nixdorf Ltd. to try to improve METAL (Adriaens and Caeyers, 1990; Thurmair, 1992), a commercial rule-based system, by manual sentence decomposition. Researchers were faced with the problem that, since most of the rules comprising the modules in METAL were designed based on simple toy sentences, the quality of the system sharply decreased when faced with longer sentences in a real-world scenario. Therefore, when testing the performance of METAL for the translation of legal texts at the Belgian Ministry of the Interior (Deprez et al., 1994), it was decided to incorporate a manual sentence decomposition module to reduce the original complexity of the sentences and boost the overall quality of the output. The decomposition module was named 'Tarzan', since it was designed with simplicity and robustness as main guidelines. In a pre-processing step,

long input sentences⁹ were manually decomposed into smaller chunks, some of which were substituted by placeholders. The placeholders indicate a certain syntacto-semantic class that was recognised by the METAL engine during the subsequent translation of the chunks.

For example, *s1* would be the placeholder for a noun phrase with semantic type ‘+human’, *s0* for a noun phrase with semantic type ‘-human’, *aa* for an adverbial complement, etc. With this technique, antecedents, subjects, direct objects and adverbial or prepositional complements could be split off in order to create shorter translation units. As an example, the sentence ‘Dans une réunion qui a duré trois heures, le directeur de la division a accepté les propositions des employés’ would be decomposed as indicated in (13):

- (13) ‘Dans une réunion’ (=‘in a meeting’)
 ‘*s0* qui a duré 3 heures,’ (= *s0* which lasted for three hours)
 ‘*aa* le directeur de la division a accepté *s0*’ (= ‘*aa* the manager of the division accepted *s0*’)
 ‘les propositions des employés ’ (= ‘the employees’ proposals’)

Experiments were conducted for the language pairs Dutch→French and French→Dutch. Although no concrete results on the overall influence of *Tarzan* on the performance of METAL were published, two of the main researchers in the project¹⁰ affirmed, when contacted in 2006, that the use of *Tarzan* was able to improve the performance of METAL to a certain extent, especially when long input sentences proved too complicated for the MT engine’s analysis module to be correctly interpreted.

Both *Tarzan* and TransBooster are attempts to improve the overall translation quality of complex sentences by sentence decomposition. However, there are a number of significant differences between both approaches:

1. The decomposition by TransBooster is fully automatic, whereas in *Tarzan*, each input sentence is chunked manually as a pre-processing step.
2. In *Tarzan*, constituents are substituted by a code which is internally recognised by METAL’s translation modules. In TransBooster, constituents are replaced by

⁹There is no data available as to the exact number of words that an input sentence had to contain in order to be eligible for decomposition.

¹⁰Geert Adriaens and Filip Deprez.

Substitution Variables that have to be translated by a baseline MT system. In other words, TransBooster is independent of the baseline MT system used while *Tarzan* was implemented specifically to be interfaced with the METAL engine.

In the late 1990s, a collaboration between the University of Southern California and Systran Ltd resulted in an experiment with a sentence-splitting algorithm to reduce the complexity of long input sentences for a Japanese→English MT system (Gerber and Hovy, 1998) Based on the assumption that shorter sentences are easier to translate due to the fact that they contain fewer ambiguities, a *Sentence Splitter* module was developed to decompose certain input sentences at the clause level This module was inserted into the translation pipeline of Systran, midway in the analysis process. Japanese input sentences were split into smaller units if the following conditions were met:

1. The original sentence is a minimum of 20 words long
2. A continuative or infinitive form verb phrase is found followed by a comma *or* a clause conjunction is found.
3. The verb phrase is not functioning as an adverbial/extended particle.
4. The resulting sentences will be at least 7 words long.

In the case of a sentence split, some resulting parts were modified by adding a replacement subject to ensure that they made up a complete, new sentence. The splitting process is demonstrated in example (14), glossed from Japanese:

- (14) Original input: 'In the future, increase of the super distance aeronautical transport which centers on between the continents can be considered for certain, can expect to 21 century beginning demand for 500-1000 supersonic transport planes with 300 seats.'
Split input: 'In the future, increase of the super distance aeronautical transport which centers on between the continents can be considered for certain. You can expect to 21 century beginning demand for 500-1000 supersonic transport planes with 300 seats.'

The results of two experiments in which human evaluators were asked to judge the readability of translations generated from both split and unsplit input did not suggest that the use of the *Sentence Splitter* module significantly improved the original unsplit output, or, to quote from the authors: '*It is not unreasonable to suspect that splitting*

sentences does not, for the current quality of Systran J-E output, make much difference in understandability' (Gerber and Hovy, 1998). They cite as possible reasons for this unexpected result. (i) the set-up of the testing procedure, (ii) possible flaws in the selection of sentence-splitting points, and (iii) the relatively low overall output quality of the baseline system.

The *Sentence Splitter* module significantly differs from TransBooster in a number of important aspects:

1. The *Sentence Splitter* module was plugged into the analysis phase of a specific commercial MT system (Systran), whereas, in our approach, the entire commercial MT system is interfaced to TransBooster with the sole purpose of returning translations from input chunks. The analysis of the original sentence, the decomposition into optimal input chunks and the recomposition of its translations are done by TransBooster itself
2. The *Sentence Splitter* module only focuses on splitting sentences at clause level.
3. Unlike in TransBooster, the decomposition of the *Sentence Splitter* module is not recursive: it stops as soon as the algorithm has identified possible clause boundaries

Note that TransBooster was conceived independently from both *Tarzan* and *Sentence Splitter*. The idea behind TransBooster originated prior to learning about the existence of the research mentioned in this section.

2.5 Summary

In this chapter, we have motivated the rationale behind TransBooster. After giving a brief overview of the most important MT paradigms, we explained the basic idea underlying our approach, namely that a recursive complexity reduction at the input side can lead baseline MT systems to improve on their own output. We compared the TransBooster approach to other MT paradigms by situating it in the three-dimensional MT model space of (Wu, 2005). Finally, we compared our approach to relevant related research.

Chapter 3

Methodology: Baseline MT Systems, Development Phases, Evaluation

3.1 Introduction

In this chapter, we outline the methodology used throughout the rest of this dissertation. In Section 3.2, we briefly describe the baseline MT systems that were interfaced with TransBooster. We provide more information on the format of the input into the decomposition algorithm in Section 3.3. Finally, in Section 3.4, we explain how the performance of TransBooster is evaluated.

3.2 Baseline MT Systems

The idea of TransBooster originated after analysing common flaws of freely available, on-line MT systems, most of which are rule-based. Therefore, as a first obvious choice, we decided to interface TransBooster to several commercial rule-based systems: LogoMedia, Systran and SDL. These systems were selected based on their relevance on the translation market (Hutchins et al , 2006), their overall quality and the availability of the language pair that we required for testing (English→Spanish). We initially experimented with a

fourth on-line MT system, PromT¹, but decided not to proceed with this system in a later stage of the project in order to scale down the experiments to a manageable size.

Initial translations were performed by accessing the systems on-line. Since all systems restrict the size of input files for on-line processing, each time a translation was needed, it was necessary to split the input into a number of smaller files, upload the files onto a web server, access the translation engines with a script executing WGET² in batch-mode and assemble the output. In order to speed up this process and to avoid occasional failures of the on-line engines, we acquired academic licences for the in-house use of LogoMedia and Systran. It was not possible to acquire the engine of SDL, so we continued with accessing the SDL engine on-line.

Despite the fact that most commercial wide-coverage MT systems are rule-based at present, it is interesting to verify the effect of a TransBooster approach on top of CBMT systems as well, since most MT research today is corpus-based. Some of the major difficulties that data-driven MT systems face (e.g. word order issues, inability to capture long-distance dependencies) relate to their lack of syntactic knowledge. Since SMT and EBMT are the two major exponents of the data-driven approach to MT, we examine in Chapter 6 whether the syntactically-driven decomposition algorithm of TransBooster is able to improve the output of an SMT and an EBMT system.

The baseline SMT system that we used is an in-house constructed phrase-based SMT system (*English*→*Spanish*) using the Giza++ alignment tool (Och and Ney, 2003), the SRI Language Modeling Toolkit (Stolcke, 2002) and the Pharaoh decoder (Koehn, 2004). The system was trained on data from the English–Spanish training section of the Europarl corpus (Koehn, 2005). More detailed information on the construction of the SMT system is provided in Chapter 6.

The baseline EBMT system that we used is the NCLT's³ marker-based MATREX system (Armstrong et al, 2006). More information about this system will be provided during the discussion of the experimental setup for the EBMT evaluation in Chapter 6.

The core components of TransBooster are language-pair independent, on the condition

¹<http://www.e-prompt.com>

²WGET is a free software package for retrieving files using HTTP, HTTPS and FTP. <http://www.gnu.org/software/wget>

³National Centre for Language Technology, Dublin City University.

that the input is parsed into a structure similar to the one used in the Penn-II Treebank.⁴ Only a limited number of modules in the program rely on language-specific material.⁵ However, for evaluation purposes, a specific language-pair had to be selected. We chose to evaluate our system on the language pair English→Spanish since (i) this commercially relevant language pair is implemented by most on-line MT systems, (ii) a large amount of training data (Koehn, 2005) is available for the construction of CBMT systems, and (iii) the developer is familiar with both languages.

3.3 Development Phases

In the first phase of the project, we used as input data to TransBooster an existing treebank resource, the Wall Street Journal (WSJ) section of the Penn-II Treebank (Marcus et al., 1994), containing about 1,000,000 words and 50,000 trees/sentences. The Penn Treebank is the largest available human parse-annotated corpus of English, and has been used as the standard test and training material for statistical parsing of English. Since the linguistic structure of the sentences in the Penn Treebank has been constructed/revised by human annotators, it is considered to be near perfect. In other words, using the parse-annotated Penn-II sentences as input data is equivalent to using a hypothetical TransBooster system with a ‘perfect’ analysis module that does not introduce any noise. Therefore, the results that we obtain for these ‘perfectly annotated’ sentences will yield a theoretical upper bound for the improvements that are possible with our approach based on automatically parsing new unannotated text.

In the second phase of the project, we experimented with a number of existing parsing methods to analyse previously unseen sentences. The resulting analysis serves as input to the decomposition algorithm developed during the first development phase. Since the output format of most state-of-the-art statistical parsers differs only slightly from the Penn Treebank annotation, the main structure of the decomposition algorithm remains valid.

The main research question here is to find out whether the best possible parser-based

⁴Current state-of-the-art Penn-II trained probabilistic parsers (Collins, 1999; Charniak, 2000, Bikel, 2002) produce this type of output structure

⁵Cf. Table F.1 in Appendix F for an overview of language-dependent vs. language-independent elements in TransBooster

analyses are good enough for TransBooster to improve translation scores with respect to the baseline systems. Or, in other words, is the TransBooster architecture resistant to the inevitable errors and noise introduced by even the best available parsers? Current state-of-the-art probabilistic parsing technology is capable of providing tree-based precision & recall scores of around 90%. We conducted experiments with (Charniak, 2000) and (Bikel, 2002), the results of which are analysed in Chapter 6.

3.4 Evaluation

In this section, we explain how the performance of TransBooster is evaluated. First, we briefly analyse the automatic evaluation metrics that will be used and explain our manual evaluation standards. We then motivate the characteristics of our test set and outline how it was constructed

3.4.1 Evaluation Metrics

During the past few years, the use of automatic evaluation metrics has become widespread in the MT community. Unlike traditional manual evaluations, usually based on a combination of accuracy and fluency (White and Connell, 1994; Hovy, 1999), automatic evaluation metrics are fast, cheap and provide an objective framework for comparison. Led by the success of the Word Error Rate metric in the evaluation of speech recognition systems, MT researchers have come up with a plethora of automatic, string-matching based, evaluation metrics in their own field: WER (Word Error Rate) (Nießen et al., 2000), RED (Akiba et al., 2001), BLEU (Papineni et al., 2002), NIST (Doddington, 2002), PER (Position independent Word Error Rate) (Leusch et al., 2003), GTM (Turian et al., 2003), the metric by (Babych and Hartley, 2004), ROUGE (Lin and Och, 2004a), METEOR (Banerjee and Lavie, 2005). All previously cited metrics have in common that they evaluate the output of an MT system against a number of reference translations, based on the rationale that the more similar an MT output is to an expert reference translation, the better it is. The individual metrics differ in the algorithms used to compute the similarity score.

Although the outcome of an automatic evaluation metric is meaningless in itself⁶

⁶That is, an absolute BLEU score of 0.23, for example, without information on the set of reference

and n -gram-based metrics have been shown to favour SMT systems over rule-based ones (Callison-Burch et al., 2006), automatic evaluation metrics are useful for MT development and comparative evaluations between MT systems of the same kind.⁷ Even though very few researchers nowadays question the usefulness of automatic MT metrics, especially for the day-to-day development of MT systems, automatic metrics are not, and were never designed to be, a *substitute* for human assessment of translation quality. The developers of BLEU, one of the earliest and best known metrics in the field, state:

‘We present this method as an automated *understudy* to skilled human judges which substitutes for them when there is need for quick or frequent evaluations.’ (Papineni et al., 2002)

Therefore, it remains indispensable to evaluate the output quality of TransBooster using human judges.

In what follows, we briefly describe the three (widely-used) automatic evaluation metrics that are used in this dissertation and explain our standards for human evaluation

3.4.1.1 BLEU

The BLEU⁸ metric (Papineni et al., 2002) compares MT output with expert reference translations in terms of n -gram statistics. The metric calculates the geometric average of a clipped unigram to 4-gram precision and applies a length penalty for translations that are too short. The details of the metric are shown in equation 3.1.

As an example⁹, consider the candidate MT output¹⁰ in (15):

(15) ‘It is a guide to action which ensures that the military always obeys the commands of the party.’

translations or the type of MT system used, is not informative about the output quality of the system.

⁷Automatic evaluation metrics have been shown to correlate with human judgements when statistical MT systems are compared (Doddington, 2002; Li, 2005).

⁸In this dissertation, we used BLEU version 1.1a.

⁹Most of the examples in this section are adapted from (Papineni et al., 2002)

¹⁰The source language is not relevant for evaluation purposes

$$BLEU = \exp \left(\sum_{n=1}^N w_n \log(p_n) - BP \right) \quad (3.1)$$

$$\text{where } p_n = \frac{\sum_i \left(\begin{array}{l} \text{the number of } n\text{-grams in sentence } i, \text{ in the translation being evaluated,} \\ \text{with a matching reference co-occurrence in sentence } i \end{array} \right)}{\sum_i \left(\begin{array}{l} \text{the number of } n\text{-grams in sentence } i, \text{ in the} \\ \text{translation being evaluated} \end{array} \right)}$$

$$w_n = N^{-1}$$

$$N = 4$$

$$BP = \max \left(\frac{L_{ref}^*}{L_{sys}} - 1, 0 \right)$$

L_{ref}^* = the number of words in the reference translation that is closest in length to the translation being scored

L_{sys} = the number of words in the translation being scored

We will calculate the BLEU score of (15) against the three human reference translations in (16):

- (16) a. 'It is a guide to action that ensures that the military will forever heed Party commands.'
 b. 'It is the guiding principle which guarantees the military forces always being under the command of the Party.'
 c. 'It is the practical guide for the army always to heed the directions of the party'

Of the 18 unigrams present in the candidate sentence (15), 17 are found in one or more of the reference translations. Therefore $p_1 = \frac{17}{18}$. Likewise, we find that for bigrams, $p_2 = \frac{10}{17}$, for trigrams, $p_3 = \frac{7}{16}$ and for 4-grams, $p_4 = \frac{4}{15}$. Also, $L_{sys} = L_{ref}^* = 18 \rightarrow BP = \max \left(\frac{L_{ref}^*}{L_{sys}} - 1, 0 \right) = 0$. Therefore

$$\begin{aligned} BLEU &= \exp \left(\sum_{n=1}^N w_n \log(p_n) \right) \\ &= \exp \left(\frac{\log(\frac{17}{18}) + \log(\frac{10}{17}) + \log(\frac{7}{16}) + \log(\frac{4}{15})}{4} \right) \\ &= 0.5045 \end{aligned}$$

It is important to mention that the n -gram precision score of a given candidate translation is clipped to the maximum of n -gram occurrences in any single reference translation to avoid overinflated n -gram scores, as is shown in (17):

- (17) Cand: the the the the the the the.
 Ref1: The cat is on the mat.
 Ref2: There is a cat on the mat.

In (17), the candidate translation would obtain a non-clipped unigram precision of 7/7. By not allowing more n -gram matches than the maximum number of n -gram occurrences in a reference translation, this precision is modified to a much more reasonable unigram precision of 2/7 for this improbable translation.

Candidate translations which are too short are penalised by subtracting a brevity penalty $BP = \max\left(\frac{L_{ref}^*}{L_{sys}} - 1, 0\right)$ from the clipped precision count. In (18), we see a candidate sentence in which $\sum_{n=1}^N w_n \log(p_n) = 0$ due to a clipped unigram to 4-gram precision of 100%. Without taking the brevity penalty of $\max\left(\frac{13}{4} - 1, 0\right) = 2.25$ into account, the BLEU score of the candidate sentence would be a ‘perfect’ score of $\exp(0) = 1$. The use of the brevity penalty reduces this number to a much more reasonable $\exp(-2.25) = 0.0056$.

- (18) Cand: This is an example.
 Ref1: This is an example of the use of the brevity penalty in BLEU

As with human judgements, scores for individual sentences can vary from judge to judge, so evaluation is normally performed on a reasonably large test set.¹¹ Since standard BLEU calculates a *geometric* average of unigram to 4-gram precision, a sentence without any 4-gram match with the reference translations, will not contribute to the overall score of the test set, despite possible successful unigram to trigram matches in the sentence. Therefore, BLEU is known to correlate better with human evaluations of *fluency* than of *accuracy* (Lin and Och, 2004b).

3.4.1.2 NIST

The NIST¹² metric (Doddington, 2002) is a variant of BLEU which uses an arithmetic average instead of a geometric average of n -gram counts, weights more heavily those n -grams that are more informative and uses an improved sentence length penalty. Details of the NIST metric are shown in equation 3.2.

¹¹BLEU scores of less than 200 sentences are rarely published.

¹²In this dissertation, we used NIST version 11a

$$NIST = \sum_{n=1}^N \left(\frac{\sum_{\substack{\text{all } w_1 \dots w_n \text{ that} \\ \text{co-occur}}} \text{Info}(w_1 \dots w_n)}{\sum_{\substack{\text{all } w_1 \dots w_n \text{ in} \\ \text{sys output}}} (1)} \right) \times BP \quad (3.2)$$

where $\text{Info}(w_1 \dots w_n) = \log_2 \left(\frac{\text{the \# of occurrences of } w_1 \dots w_{n-1}}{\text{the \# of occurrences of } w_1 \dots w_n} \right)$

$$BP = \exp \left(\beta \log^2 \left[\min \left(\frac{L_{sys}}{\bar{L}_{ref}}, 1 \right) \right] \right)$$

β = a factor to make $BP = 0.5$ when the # of words in the system output is $\frac{2}{3}$ of the average # of words in the reference translation

$N = 5$

\bar{L}_{ref} = the average number of words in a reference translation, averaged over all reference translations

L_{sys} = the number of words in the translation being scored

The informativeness of an n -gram is expressed by its information gain $\text{Info}(w_1 \dots w_n)$, which is higher for n -grams that occur less frequently. For example, consider the imaginary one-sentence corpus in (19)

(19) 'The white man in the white truck followed the white rabbit in San Francisco'.

The information gain of a collocation as 'San Francisco' with respect to the unigram 'San' is $\text{Info}(\text{San Francisco}) = \log_2 \left(\frac{1}{1} \right) = 0$, since 'San' and 'Francisco' always co-occur in the corpus. The information gain of the bigram 'white rabbit' is $\text{Info}(\text{white rabbit}) = \log_2 \left(\frac{3}{1} \right) = 1.58$. Therefore, a match in a reference translation of the more informative (or less likely) bigram 'white rabbit' will contribute more to the overall NIST score than a match of the less informative (or more likely) bigram 'San Francisco'. A downside to this approach is that certain valuable higher order n -gram matches will not contribute to the NIST score if their information gain is zero, which is not unlikely. Zhang et al. (2004) show that 80% of the NIST score for a typical MT system comes from unigram matches, the main reason being that the information gain of lower-order n -grams is typically higher than the information gain of higher-order n -grams. Therefore, NIST is known to correlate better with human evaluations of *accuracy* than of *fluency*.

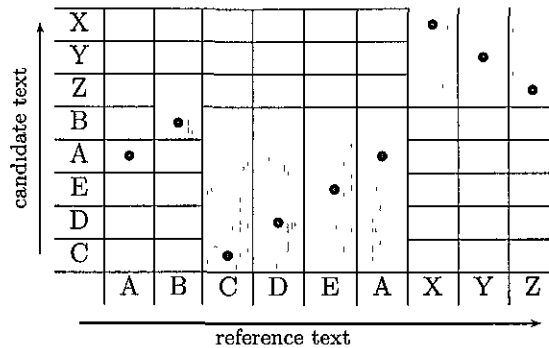


Figure 3.1: Bitext grid illustrating the relationship between an example candidate translation and its corresponding reference translation. Each bullet or ‘hit’ indicates a word contained in both the candidate and reference texts.

3.4.1.3 GTM

The General Text Matcher (GTM¹³) metric (Turian et al., 2003) was developed to express MT evaluation in terms of the standard measures of precision and recall, which according to the authors, are more intuitive than BLEU or NIST. For a given set of candidate items C and a set of reference items R , precision and recall are defined in (3.3) and (3.4) respectively:

$$precision(C|R) = \frac{|C \cap R|}{|C|} \quad (3.3)$$

$$recall(C|R) = \frac{|C \cap R|}{|R|} \quad (3.4)$$

The precision/recall of a translation with respect to a reference translation can be graphically represented as a bitext grid as in Figure 3.1, in which each bullet or ‘hit’ represents a word in common between the reference translation on the X-axis and the candidate translation on the Y-axis. In order to avoid double counting¹⁴, (Turian et al., 2003) replace the concept of a ‘hit’ by a ‘match’, defined as the subset of hits in the grid, such that no two hits are in the same row or column. In Figure 3.1, *matches* are represented by hits in a shaded area. They then define the precision/recall of in terms of

¹³In this dissertation, we used GTM version 1.2

¹⁴For example, there are two hits for block A, but only one is relevant to calculate precision/recall.

the Maximum Match Size (MMS) between candidate and reference texts

$$precision(C|R) = \frac{MMS(C, R)}{|C|} \quad (3.5)$$

$$recall(C|R) = \frac{MMS(C, R)}{|R|} \quad (3.6)$$

As an example, the MMS for the grid in Figure 3.1 is 8 (calculated by summing the sizes for the individual smaller matchings of 1, 4 and 3, as indicated by the shaded areas in the grid), the length of the candidate text is 8 and the length of the reference text is 9, so precision in this case is $8/8 = 1.0$, whereas recall is $8/9 = 0.89$.

In order to reward correct word order in addition to individual matches, contiguous sequences of matching words ('runs') are weighted according to their length, so that the MMS between candidate and reference texts is redefined as in (3.7):

$$MMS = \sqrt{\sum_{runs} length(run)^2} \quad (3.7)$$

After identifying the runs (hits occurring diagonally adjacent in the grid running parallel to the main diagonal) and corresponding aligned blocks of the two candidate texts, as indicated by the shaded areas in Figures 3.2(a) and 3.2(b), we can use the formula in equation 3.7 to calculate the MMS for each candidate text and their corresponding precision and recall scores. Looking at Figure 3.2, the MMS for the candidate in Figure 3.2(a) is $\sqrt{1^2 + 4^2 + 1^2 + 1^2 + 1^2} \approx 4.5$ and $\sqrt{1^2 + 4^2 + 3^2} \approx 4.9$ for the candidate in Figure 3.2(b), giving Figure 3.2(a) precision of $4.5/8 = 0.5625$ and recall of $4.5/9 = 0.5$, whereas Figure 3.2(b) scores a higher precision of $4.9/8 = 0.6125$ and higher recall of $4.9/9 = 0.5445$, reflecting the higher quality of this particular candidate text.

The GTM metric can easily be extended to multiple reference translations by concatenating the various reference texts into a single grid with minor adaptations (Turian et al., 2003). The final GTM score is expressed as the harmonic mean or F-score (van Rijsbergen, 1979) of precision (P) and recall (R) in equation 3.8:

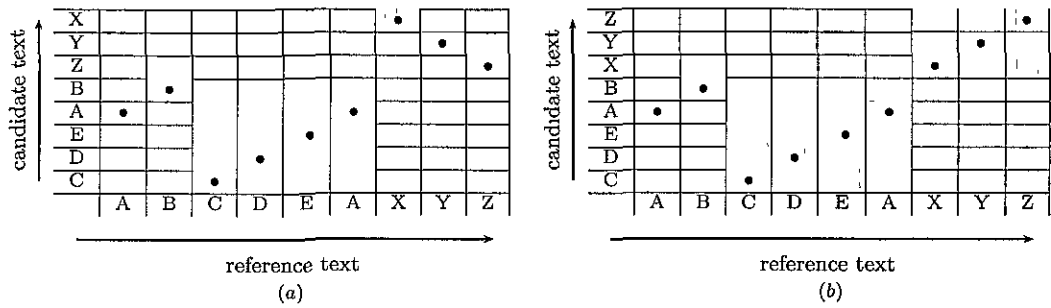


Figure 3.2: Bitext representing two different candidate texts for the same reference text. The MMS in Equation 3.7 rewards the better word order in candidate text (b) by weighting each contiguous sequence of matching words by their length, which is indicated by the greater surface of shaded area in (b)

$$GTM = \frac{2PR}{P + R} \quad (3.8)$$

3.4.1.4 Statistical Significance

The statistical significance of the results mentioned in this thesis that were obtained by the previously mentioned metrics was established in each case in a 95% confidence interval using bootstrap resampling on 2000 resampled test sets (Davison and Hinkley, 1997). In cases where the obtained results were found *not* to be statistically significant, an explanation is provided. If no explicit mention of statistical significance testing is made, the results are statistically significant.

3.4.1.5 Manual Evaluation

In a recent study on manual and automatic evaluation of Machine Translation (Koehn and Monz, 2006), the suggestion was made to replace the traditional *absolute* human evaluations¹⁵ by a *relative*, ranked evaluation for comparative purposes. This is motivated by the fact that it is often difficult for human judges to adhere to the same criteria while evaluating a test suite and that, on an absolute scale (e.g. 1-5), they tend to choose the ‘safe’ middle value (e.g. 3), neglecting smaller but still important differences between translations. Since we are interested in the performance of TransBooster with respect to

¹⁵Output sentences are usually graded for accuracy and fluency on an absolute scale, for example, from 1 (very poor) to 5 (perfect).

the individual baseline systems, we decided to use this new comparative, *relative* evaluation method. Therefore, when conducting the evaluations reported in Chapters 6 and 7, the human judges were asked to select, for each sentence pair $\langle \textit{TransBooster output} - \textit{Baseline MT output} \rangle$, the better translation (if any), both in terms of accuracy and fluency.

3.4.2 Experimental Set-up

In order to evaluate the output quality produced by TransBooster, we constructed an 800-sentence test set (with sentence length between 1 and 54 words, ave. 19.75 words) from Section 23 of the Penn-II Treebank. This test set is composed of the 700 sentences in the PARC-700 dependency bank (King et al., 2003), the 105 sentences in the DCU-105 dependency bank (Cahill et al., 2004) and 17 sentences, randomly selected from Section 23 of the Penn-II Treebank to make up for overlapping sentences in the PARC-700 and DCU-105. We preferred to join 2 previously existing test sets over constructing an entirely new set because of the wide acceptance and usage of these test sets in the dependency parsing community.

In order to construct a set of gold standard human reference translations for the automatic MT evaluation metrics, we had the 800-sentence test set translated into Spanish by 4 native translators who had graduated from the School of Applied Language and Intercultural Studies (SALIS) at Dublin City University. All 4 translators were presented with 200 input sentences, randomly selected from the test set. We had previously translated each of these sentences by one out of 4 MT engines (LogoMedia, Systran, SDL and PromT), in a random order. This MT output was also presented to the translators. The translators were asked to use (parts of) the MT output if considered useful and to evaluate the quality of the Machine Translation by giving each sentence a score between 5 (very useful) and 1 (useless), as is shown in Table 3.1.

Although most human evaluations of Machine Translation involve computing an average between two scores, one score measuring the quality of the target language sentence (fluency), the other measuring the semantic similarity between output and input (accuracy) (Hovy et al., 2002), we chose to use only one score so as not to burden the translators

Score	Meaning	Criteria
5	very useful	'I copied the entire translation and made minor changes.'
4	useful	'I found most elements in the translation useful.'
3	neutral	'I found some elements in the translation useful.'
2	not really useful	'I found few elements in the translation useful.'
1	useless	'I found nothing or almost nothing in the translation useful.'

Table 3.1: Extract from the instructions for the translation of the test set.

and distract them from their main task (to produce a perfect translation of the input sentence, with or without the help of MT). The score we used roughly measures the required amount of post-editing, which is a practical measure of quality and includes both concepts of accuracy and fluency. Although the main goal was to obtain 'perfect' human translations of the test set, the MT evaluation also gave us an initial idea of the strength of the different MT engines.

To ensure that all translators would perform this task in a coherent fashion and to facilitate the retrieval of the results, we built an interactive web page that the participants could access at any time to do the translations and review/modify their input if necessary. Part of this web page is displayed in Figure 3.3.

Given that, in many cases, several correct translations exist for a source language sentence, it is preferable to provide automatic MT evaluation metrics with more than one reference translation. In (Zhang and Vogel, 2004), the authors investigate the effect of increasing the number of reference translations on the precision of several automatic MT evaluation metrics. As is to be expected, they find that a higher number of reference translations results in a narrower confidence interval, i.e. it increases the precision of the metrics. They also investigate the effect of increasing the testing data size on the precision of the metrics. Interestingly, they find that adding an additional reference translation compensates for the effects of removing 10–15% of the testing data on the confidence interval. Therefore, although both increasing the size of the testing data as well as using more reference translations increases the precision of the evaluation metrics, it seems more cost-effective to use more test sentences than to increase the number of reference translations.

In other words, the confidence interval of the evaluation metrics narrows down more

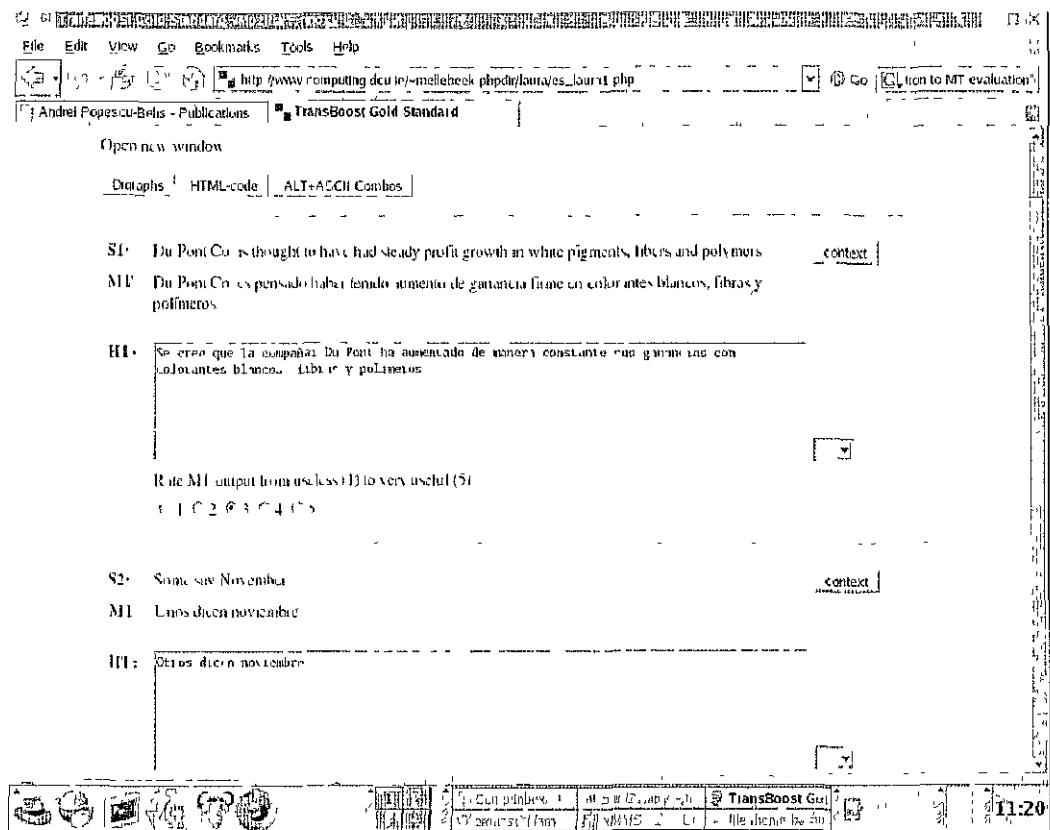


Figure 3.3. A section of the web page for translators to construct the gold standard reference translations.

by using 800 test sentences with one reference translation, than, for example, 200 test sentences with four reference translations. This explains why, faced with the question whether to maximise either the test data size or the number of reference translations given a fixed budget for translations, we chose the first alternative. Moreover, the use of a larger test set allows us to evaluate a larger variety of syntactic phenomena.

3.5 Summary

In this chapter, we have introduced the baseline MT systems used in this thesis and have explained how we will evaluate the performance of TransBooster with respect to these systems. The baseline systems are three widely-used commercial RBMT systems, one in-house constructed SMT system, and one research-oriented EBMT system. The

performance of TransBooster will be measured on an 800-sentence test set extracted from Section 23 of the Penn-II Treebank, based on three standard automatic evaluation metrics and a comparative manual evaluation

Chapter 4

TransBooster Architecture: Outline

4.1 Introduction

This chapter introduces the concepts necessary to understand the technical details of TransBooster, which are explained in depth in Chapter 5. There are two main sections in this chapter. Section 4.2 contains an outline of the TransBooster architecture and illustrates the application of parts of the algorithm on several example sentences. In Section 4.3, we introduce the concept of *Substitution Variables* and report the results of a preliminary experiment conducted to determine the suitability of various Substitution Variable schemata.

4.2 Outline

This section contains an outline of the basic TransBooster architecture and introduces the associated terminology that will be used throughout the rest of this dissertation.

TransBooster takes as input a Penn Treebank-like syntactic analysis. In a first step, the input tree is flattened for further processing (Section 4.2.1). This is done by chunking the input tree into a *pivot* (Section 4.2.2) and a number of *satellite* chunks (Section 4.2.3). In the next step, the satellite chunks are substituted with simple replacement strings that reduce the complexity of the original input (Section 4.2.4). This simplified string is sent

to the baseline MT engine for translation, which renders the translation of the pivot and the location of the satellites in target. If the identified satellite chunks are deemed simple enough for translation, they are embedded in a context template mimicking the original context and translated by the baseline MT system (Section 4.2.5). The entire process is recursively applied to each chunk considered too complex for direct translation (Section 4.2.6). In a final step, after the entire input string has been decomposed into N chunks $C_1 \dots C_N$ and all chunks have been translated in simplified contexts, the output is formed by recombining the chunk translations

We will illustrate each stage in the process with the example sentence in (20):

(20) 'The chairman, a long-time rival of Bill Gates, likes fast and confidential deals.'

The translation (English→Spanish) of (20) by Systran is (21):

(21) 'El presidente, rival de largo plazo de Bill Gates, gustos ayuna y los repartos confidenciales.'

In (21), the MT system erroneously analyses the verb 'likes' as a noun (→'gustos') and identifies the adjective 'fast' as a verb (→'ayuna'), which renders the output unintelligible. In the following sections, we will demonstrate how TransBooster can help the baseline MT system improve its own output translation.

4.2.1 Flattening Penn-II Trees into TransBooster Trees

In order to prepare an input sentence for processing with TransBooster, the Penn-II-style tree for that string is flattened into a simpler structure consisting of a *pivot* and a number of *satellites*. The pivot of an input constituent consists of the grammatical head of the constituent but can optionally contain additional lexical items in cases where we consider it necessary to treat the head and the additional items as a single unit for safe translation (cf. Section 4.2.2). Basically, the pivot is the part of the input string that has to remain unaltered during the decomposition process. The expression *satellites* is an umbrella term for the pivot's argument and adjunct constituents.

After flattening the input tree into a TransBooster tree, we obtain the structure in Figure 4.1. This structure is the input to the decomposition algorithm.

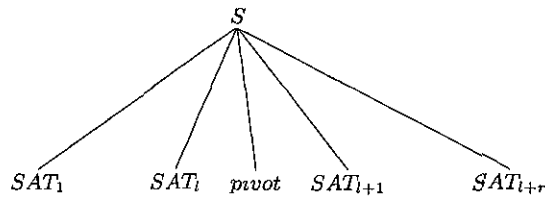


Figure 4.1: Flattening a Penn-II tree into a TransBooster tree. l = number of satellites to left of pivot r = number of satellites to right of pivot.

As an example, consider the Penn-II tree in Figure 4.2. After finding the pivot 'likes' (cf. Section 4.2.2) and locating the satellites 'the chairman, a long-time rival of Bill Gates' and 'fast and confidential deals' (cf. Section 4.2.3), we obtain the flattened structure in (22), graphically represented in Figure 4.3.

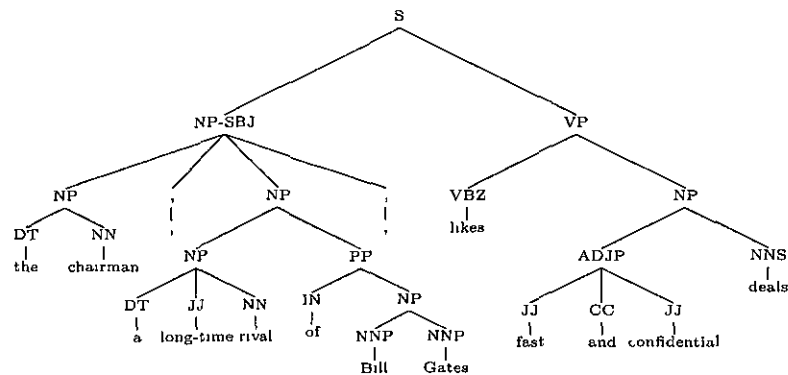


Figure 4.2: Penn-II tree representation of 'The chairman, a long-time rival of Bill Gates, likes fast and confidential deals'

(22) [The chairman, a long-time rival of Bill Gates]_{SAT₁} [likes]_{pivot} [fast and confidential deals]_{SAT₂}

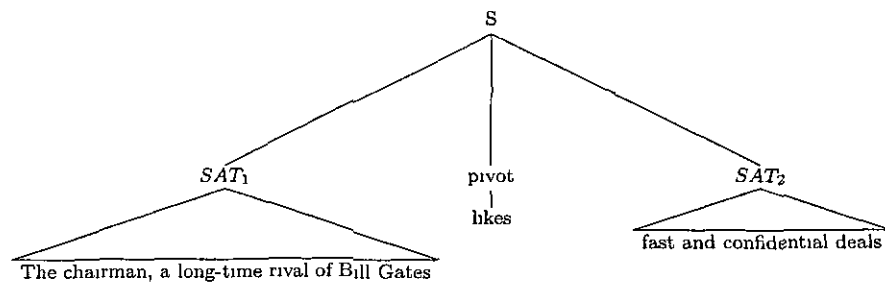


Figure 4.3: Flattened TransBooster tree obtained from Penn-II structure in Figure 4.2

4.2.2 Finding the Pivot

In order to identify the pivot of the input chunk, we first compute the chunk's head. We use the head-finding rules of (Cahill, 2004), which are an adaptation of the head-lexicalised grammar annotation scheme of (Magerman, 1995) and (Collins, 1999). These rules identify the head of a constituent by traversing the list of its daughter nodes from left to right (head-initial) or right to left (head-final) and try to match each daughter node to a previously established list of head candidates¹

The pivot of a local tree is often identical to the string formed by the terminal nodes dominated by its head, but in certain cases, in addition to the head, some of its rightmost neighbours are included, where we consider it too dangerous to translate either part out of context. An example is the use of auxiliaries, as in Figure 4.4. Here the pivot extracted by TransBooster is 'might have to buy'.

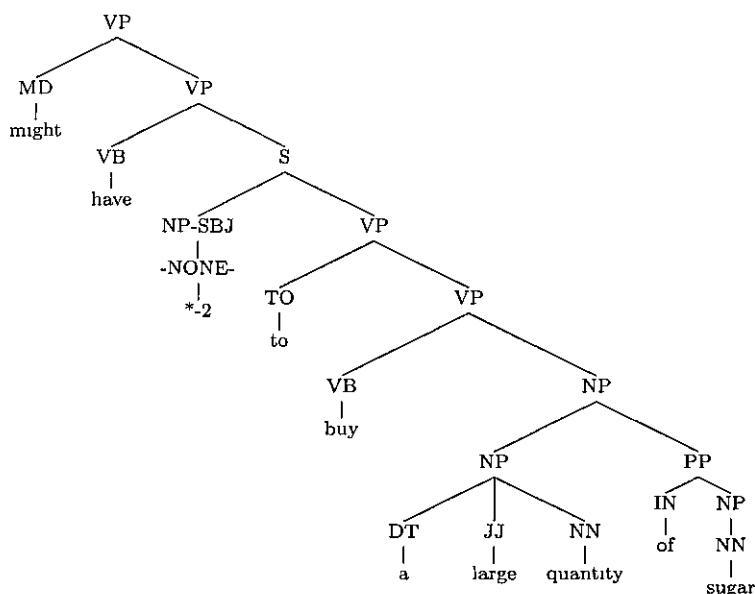


Figure 4.4. Penn-II tree representation of 'might have to buy a large quantity of sugar'

Another example is an ADJP whose head dominates a PP, as in Figure 4.5. Here the pivot established is 'close to'

¹The head-finding rules are explained in more detail in Section 5.2.1 on page 69

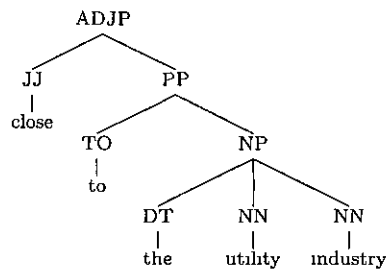


Figure 4.5: Penn-II tree representation of ‘close to the utility industry’

4.2.3 Locating Satellites

We have explained how the strings submitted to the MT system are comprised of pivots and satellites, the latter being an umbrella term for arguments and adjuncts. In this thesis, we broaden the traditional notion of the term ‘argument’ to those nodes that are required for the correct (or, at any rate, safe) translation of the string dominated by the parent node. The distinction between arguments and adjuncts is essential, since nodes labelled as adjuncts can be safely omitted in the SL string that we submit to the baseline MT system (Cf. Section 4.2.4 for more details).

For example, in (20), the strings ‘the chairman, a long-time rival of Bill Gates’ and ‘fast and confidential deals’ are arguments of the pivot ‘likes’ since neither of the strings can be left out in the SL string submitted to the baseline MT system to ensure a correct translation of the pivot ‘likes’. One of the strings that TransBooster will construct for this purpose is ‘The chairman likes deals’. On the other hand, when treating ‘the chairman, a long-time rival of Bill Gates’, the apposition ‘a long-time rival of Bill Gates’ can be safely left out in the string submitted to the MT system. The omission of adjuncts is a simple and safe method to reduce the complexity of the SL candidate strings. Additional strategies for reducing the complexity of a sentence involve substituting simpler but syntactically similar elements for chunks (Cf. Section 4.2.4 for more details).

Our procedure for argument/adjunct location is based on the argument/adjunct-finding heuristics in the algorithm used by Hockenmaier (2003) to transform the phrase-structure trees in the Penn Treebank into a corpus of CCG derivations and is explained in more detail in Section 5.2.3.

4.2.4 Skeletons and Substitution Variables

Once the original input tree has been flattened into a TransBooster tree and the pivot and satellites have been identified, in a next step the satellites are substituted with simple replacement strings that reduce the complexity of the original input. We will refer to these replacement strings as Substitution Variables (SVs), which are treated in detail in Section 4.3. The objectives of SVs are twofold:

1. They reduce the complexity of the original satellites, which can lead to an improved translation of the pivot.
2. They are used to track the location of the translation of the satellites in target.

By replacing the satellites in Figure 4.1 with their SVs, we obtain (23):

$$(23) \quad [SV_{SAT_1}] \dots [SV_{SAT_l}] \textit{pivot} [SV_{SAT_{l+1}}] \dots [SV_{SAT_{l+r}}]$$

where SV_{SAT_i} is the simpler string substituting SAT_i ($1 \leq i \leq l+r$)

TransBooster sends the simplified string (23) to the baseline MT system, which produces the output in (24):

$$(24) \quad [SV'_{SAT_1}] \dots [SV'_{SAT_l}] \textit{pivot}' [SV'_{SAT_{l+1}}] \dots [SV'_{SAT_{l+r}}]$$

Alternatively, some permutation of the elements in (24) may be derived, as the position of the translation SV'_{SAT_i} does not necessarily have to be identical to the position of SV_{SAT_i} in the source. If the translation of each of the SVs is known in advance, the string in (24) can be used (i) to extract the translation of the pivot \textit{pivot}' , and (ii) to determine the position of the translation of the satellites SAT_i in target.

It is important to stress the difference between SVs for arguments and adjuncts. Leaving out adjunct satellites in (23) will not affect the translation of the rest of that sentence, while argument satellites must always appear linked to their head and sister arguments.

The translations in (25) illustrate the fact that the argument structure of a pivot has to be kept intact at all times to retrieve the correct translation of the pivot. All input chunks are translated by LogoMedia from English→Spanish.

- (25) a. 'The man relies on the woman' → 'El hombre depende de la mujer'.
 b. 'The man relies' → *'Los hombre relies'.
 c. 'on the woman' → 'sobre la mujer'.
 d. 'The man relies' + 'on the woman' → *'Los hombre relies sobre la mujer'.

In (25), the original translation of 'The man relies on the woman' is correct. The omission of the argument 'on the woman' leads to a nonsensical translation of 'The man relies' (→*'Los hombre relies'), in which 'relies' is treated as an unknown word by Logo-Media and the article 'the' is erroneously translated in plural. The example shows that it is essential to keep the head's entire argument structure list intact when simplifying a sentence.

Since adjuncts have no influence on the translation of the pivot, the goal of adjunct SVs is only to track the translation of the adjunct in target, while argument SVs are used, (i) to embed the pivot in a simplified context which can lead to an improvement in the translation of the pivot, and (ii) to track the location of the translated arguments in target. Subsequently, the formula in (23) has to be refined to account for the differences between argument SVs and adjunct SVs.

By replacing the argument satellites in Figure 4.1 with their SVs and leaving out the adjuncts, we obtain (26)

$$(26) \quad [SV_{ARG_1}] \dots [SV_{ARG_l}] pivot [SV_{ARG_{l+1}}] \dots [SV_{ARG_{l+r}}]$$

where SV_{ARG_i} is the simpler string substituting ARG_i ($1 \leq i \leq l+r$).

We will refer to (26) as the *argument skeleton*. TransBooster sends the argument skeleton to the baseline MT system, which produces the output in (27), or some permutation of it:

$$(27) \quad [SV'_{ARG_1}] \dots [SV'_{ARG_l}] pivot [SV'_{ARG_{l+1}}] \dots [SV'_{ARG_{l+r}}]$$

where SV'_{ARG_i} is the translation of SV_{ARG_i} by the baseline MT system.

Since the translation of the argument SVs can be determined in advance, the translation of the argument skeleton, namely (27), will yield (i) the translation of the pivot in (26) as *pivot'*, and (ii) the location of the translation of the arguments in target, SV'_{ARG_i} .

In order to track the location of the translation of the adjuncts in target, we add the adjunct SVs one by one to the argument skeleton in (26). For N different adjuncts in

the input string, this will yield N different strings, which are schematically represented in (28):

$$(28) \quad [SV_{ARG_1}] [SV_{ADJ_1}] \dots [SV_{ARG_l}] [pivot] [SV_{ARG_{l+1}}] \dots [SV_{ARG_{l+r}}]$$

$$\dots$$

$$[SV_{ARG_1}] \quad [SV_{ARG_i}] [SV_{ADJ_j}] pivot [SV_{ARG_{l+1}}] \dots [SV_{ARG_{l+r}}]$$

$$\dots$$

$$[SV_{ARG_1}] \dots [SV_{ARG_i}] pivot [SV_{ARG_{l+1}}] \dots [SV_{ARG_{l+r}}] [SV_{ADJ_N}]$$

where SV_{ARG_i} is the simpler string substituting ARG_i ($1 \leq i \leq l+r$) and SV_{ADJ_j} is the simpler string substituting ADJ_j ($1 \leq j \leq N$).

We will refer to these N different strings as *adjunct skeletons*. As with the argument skeleton, TransBooster sends each of the N adjunct skeletons to the baseline MT system and, based on the already known translation of SV_{ADJ_j} , tries to establish the location of each of the adjuncts in target.

Argument Skeleton: example

By replacing the argument satellites 'The chairman, a long-time rival of Bill Gates' and 'fast and confidential deals' by the argument SVs 'The chairman' and 'deals' in the flattened TransBooster tree of the example sentence in (22) on page 41, we obtain the argument skeleton in (29):

(29) 'The chairman likes deals.'

We retrieve the translation of the pivot by submitting this skeleton to the baseline MT system and subtracting the known translations of the SVs.² For example, the translation of (29) from English→Spanish by Systran is (30):

(30) 'El presidente tiene gusto de repartos.'

If we subtract the known translations 'El presidente' and 'repartos', we obtain the translation 'tiene gusto de' for the pivot 'likes', as well as the position of the translations of the arguments in target. 'tiene gusto de' is markedly better than the erroneous 'gustos', the original translation produced by Systran in (21) on page 40. The reason for this improvement is that the reduction in syntactic complexity has undone the deficient homograph

²cf. Section 4.3.2 on page 54 on how to determine the translation of SVs in advance

resolution of the word ‘likes’ by the baseline MT system’s analysis module in the original, full sentence; where in (20) on page 40, it was wrongly analysed as a noun, in the simpler string (29), the analysis module is able to correctly identify it as a verb.

Adjunct Skeleton: example

In order to track the position of the adjunct ‘a long-time rival of Bill Gates’ in target, we substitute the chunk with the SV ‘a rival’, which is inserted in the argument skeleton in (29), leading to (31)

(31) ‘The chairman, a rival, likes deals.’

The translation of (31) from English→Spanish by Systran is (32):

(32) ‘El presidente, rival, tiene gusto de repartos.’

Since we know the translation of the argument skeleton (30) and have previously defined the translation of the SV ‘a rival’, it is possible to determine the location of the translation of the SV, which will render the location of the adjunct chunk ‘a long-time rival of Bill Gates’ in target.

4.2.5 Translating Satellites: Context

Our approach is based on the idea that by reducing the complexity of the original context, the baseline MT system is more likely to produce a better translation of the input chunk C_i than if it were left intact in the original sentence, which contains more lexical, syntactic and semantic ambiguities. In other words, we try to improve on the translation C'_i of chunk C_i by the baseline MT engine through input simplification.

While simplifying the original sentence structure, it is important not to translate individual chunks out of context, since this is likely to produce a deficient output due to inappropriate lexical selection and boundary friction. Boundary friction is a well-known phenomenon in EBMT where the recombination of several partial translations, extracted from a bitext corpus, can give rise to conflicting grammatical information in the output. For example, if in (33), the translation for ‘man’ is simply replaced with the translation

for 'woman' in the example sentence 'Ce vieil homme est mort.', the erroneous 'Ce vieil femme est mort.', would be produced (Somers, 2003).

- (33) That old man has died ~ Ce vieil homme est mort
 man ~ homme.
 woman. ~ femme.
 That old woman has died → *Ce vieil femme est mort.

The correct translation of 'That old woman has died' is 'Cette vieille femme est morte', in which the determiner 'ce', the adjective 'vieil', and the past participle 'mort' acquire the feminine gender ('cette', 'vieille', 'morte') through agreement with 'femme'.

The example illustrates the importance of ensuring that each chunk is translated in a simple context that, as much as possible, mimics the original, while at the same time reducing the overall size and complexities of the original input. After embedding the candidate chunk into the context, the entire string is submitted to the baseline MT system, as shown in (34)

$$(34) \quad [\text{context}]_i [C_i] \rightarrow [\text{context}]'_i [C'_i]$$

If we can determine the translation of the context template beforehand, it is trivial to extract C'_i from the output string.

We make use of two different types of context templates. The first type is a *Static Context template*, a previously established template, the translation of which is known in advance. The second type is a *Dynamic Context template*: a reduced version of the original context; the translation of which has to be determined at run-time.

Static Context templates mimic the syntactic characteristics of the original context, but contain different words than the ones used in the original sentence. Consider the example sentence in (35):

- (35) 'The bankruptcy of one of our most important customers hasn't had any impact on us.'

If the chunk 'any impact on us' is translated (English→Spanish) by LogoMedia, out of context, as is shown in (36), the MT system misanalyses 'impact' as a verb, which leads to the erroneous translation *'ninguno tiene un impacto sobre nosotros' (= 'nobody has

any impact on us’). If, on the contrary, we insert the chunk into a simple static context template that mimics the direct object position of the chunk (‘The man is not eating’), LogoMedia produces the correct translation ‘ningún impacto sobre nosotros’, even if the context template in this case does not share any semantic characteristics of the original.

- (36) a. ‘any impact on us.’ → *‘ninguno tiene un impacto sobre nosotros.’
 b. ‘[The man is not eating] any impact on us.’ → ‘El hombre no está comiendo ningún impacto sobre nosotros’

While this method is effective for simple cases, as shown above, it is easy to see that successful translation retrieval with template insertion relies heavily on lexical information in the source language. Changing the original context excessively might split idiomatic constructions or undo agreement links in source and lead to erroneous translations instead of improvements. In addition, if the MT system relies on semantic information in order to generate translations, simple syntactic insertion templates might not be sufficient to ensure a correct translation. Therefore, a more robust alternative to Static Context templates is to maintain the translation candidate chunk embedded in a simplified form of its original context, which we will refer to as a *Dynamic Context* or a *Minimal Sufficient Context*. A Dynamic Context is sufficient for correct translation because its syntactic and semantic content is sufficient to ensure a correct translation of the candidate chunk. It is minimal because all redundant elements (adjuncts) have been removed.

In (37), the input chunk ‘fast and confidential deals’ is embedded in the Dynamic Context ‘[The chairman likes]_C’, which is a simplification of the original ‘The chairman, a long-time rival of Bill Gates, likes’. This reduction in complexity helps Systran (English→Spanish) to improve the translation of the input chunk from the erroneous ‘ayuna y los repartos confidenciales’ to the correct ‘repartos rápidos y confidenciales’.

- (37) The chairman, a long-time rival of Bill Gates, likes [fast and confidential deals] → ‘El presidente, rival de largo plazo de Bill Gates, gustos [ayuna y los repartos confidenciales].’
 [The chairman likes]_C [fast and confidential deals] → [El presidente tiene gusto de]_C
 [repartos rápidos y confidenciales].

We have seen in (30) on page 46 that the reduction in syntactic complexity by SV

substitution helps to improve the translation of the pivot. Here, the reduction in syntactic complexity of the original context helps to improve the translation of the satellites.

The trade-off in using the more similar Dynamic Contexts instead of predefined Static Context templates is that, contrary to the use of Static Context templates, the retrieval of the translated candidate chunk is no longer trivial, since we do not know the translation of the Dynamic Context in advance. It is possible, however, as we will show in Section 5.2.5, to retrieve the translation of the candidate chunk with a high degree of certainty in most cases by translating the Dynamic Context template at run-time.

4.2.6 Recursion

The TransBooster decomposition algorithm starts at the root node of the flattened Penn-II syntactic annotation tree representing the input string and examines each satellite chunk SAT_i . If SAT_i is deemed simple enough for translation, it is embedded in a simplified context as described in Section 4.2.5 and sent off to the baseline MT system for translation. If SAT_i is deemed too complex for translation, the TransBooster procedure is recursively applied to SAT_i , i.e. the satellite chunk itself is decomposed into a pivot and satellites, which in turn are examined for translatability. In other words, TransBooster recursively decomposes the original input string into a number of optimal chunks, each of which is translated in a simplified context. The recursive nature of the decomposition procedure is graphically represented in Figure 4.6.

The conditions to determine the translatability of a candidate chunk depend on the number of lexical items contained in the chunk (cf. Section 5.2.6) and the MT system used. It was determined empirically, for each different baseline MT system, by tuning the program parameter `p_ChunkLength`, as will be further explained during the discussion of experimental results in Chapter 6. After recursively decomposing the input sentence into a number of optimal chunks and sending these chunks to the baseline MT engine in a reduced context, the output sentence is formed by combining the retrieved chunk translations. This recombination is possible since we have kept track of the relative position of each chunk with respect to its pivot by using SVs as described in Section 4.2.4.

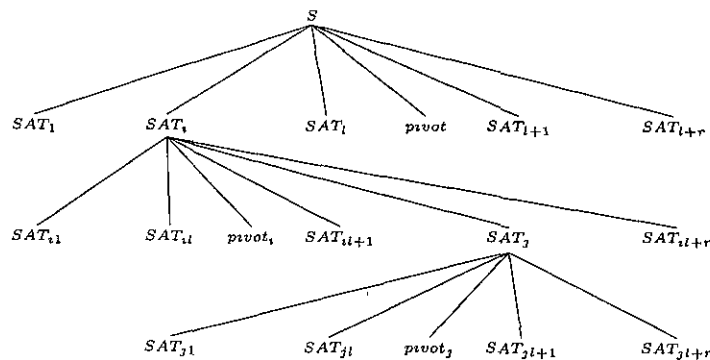


Figure 4.6: The recursive nature of the TransBooster decomposition. each satellite chunk SAT_i is decomposed until only optimal chunks remain.

4.2.7 A Worked Example

In this section, we will illustrate the entire TransBooster process on the Penn-II sentence in (38)

(38) 'Imperial Corp., based in San Diego, is the parent of Imperial Savings & Loan'

The baseline MT system is LogoMedia, the language pair English→Spanish. The output of the example sentence by the baseline system is displayed in (39)

(39) 'Imperial Corp., Fundar en San Diego, ser el padre de Savings & Loan imperial.'

There are two major problems in this translation (i) 'based' is erroneously translated as 'Fundar' (= 'to found'), and (ii) 'ser' (= 'to be') is not conjugated.

The input to the decomposition algorithm is the Penn-II tree in Figure 4.7:

Step 1

The algorithm finds the pivot 'is' and the satellites 'Imperial Corp, based in San Diego,' and 'the parent of Imperial Savings & Loan'. This leads to the flattened structure in Figure 4.8:

TransBooster replaces both argument satellites by a (in our example, static) SV ('John' and 'the boy' respectively) and sends the argument skeleton in (40) to the baseline MT engine. Since we know the translation of the SVs ('John' and 'el niño'), it is possible to

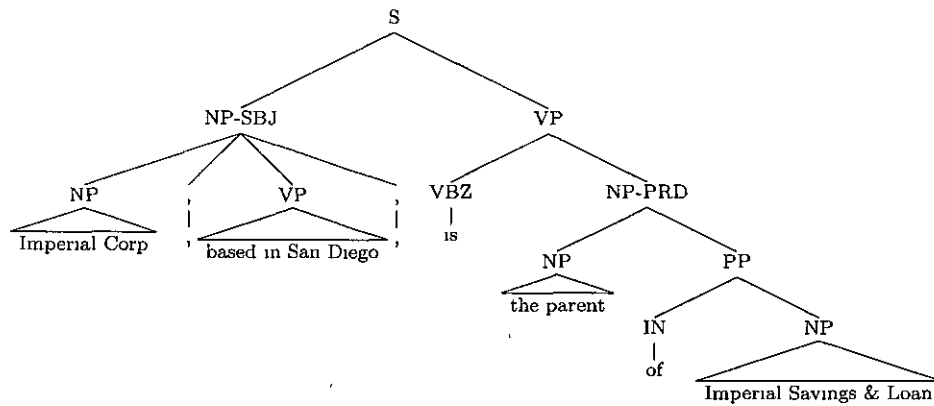


Figure 4.7: Penn-II tree representation of 'Imperial Corp, based in San Diego, is the parent of Imperial Savings & Loan'

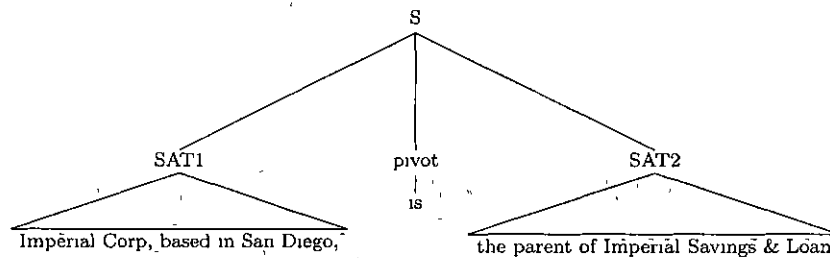


Figure 4.8: TransBooster tree representation of (4.7).

'extract the translation of the pivot ('es') and locate the position of the argument satellites in target.

(40) '[John] is [the boy]' → '[John] es [el niño]'

Step 2

Next, the first satellite ('Imperial Corp., based in San Diego') is submitted to the decomposition algorithm, which finds the pivot 'Imperial Corp.' and adjunct satellite 'based in San Diego'. Since the presence of the adjunct is not required for a safe translation of the pivot, the argument skeleton consists only of the pivot 'Imperial Corp.', which the baseline MT system translates as 'Imperial Corp.'. In order to find the location of the translation of the adjunct, we replace the adjunct with the syntactically similar SV 'made in China', which leads to the adjunct skeleton in (41). From the translation of this skeleton we deduce the position of the adjunct in target

(41) 'Imperial Corp., made in China' → 'Imperial Corp , hacer en China.'

Step 3

The algorithm now investigates the adjunct chunk 'based in San Diego' and decides that it is simple enough for translation. Since it is necessary to embed the chunk in a context that mimics the original, the chunk is preceded by a proper noun template 'John', the translation of which is known in advance. This leads to the string in (42), which is translated by the baseline MT system. From the output, we deduce the translation of the chunk: 'ubicado en San Diego'.

(42) 'John, based in San Diego.' → 'John, ubicado en San Diego'

Step 4

After the first argument satellite in Figure 4:7 has been decomposed and translated, the algorithm focuses on the second satellite ('the parent of Imperial Savings & Loan'), which is decomposed into the pivot 'the parent of' and the argument satellite 'Imperial Savings & Loan'. Note that the pivot in this case is more comprehensive than the grammatical head of the NP 'the parent of Imperial Savings & Loan'. The reason for this is that we want to prevent the preposition 'of' from being translated separately from its head 'the parent' due to the idiomaticity of preposition translation. The argument satellite 'Imperial Savings & Loan' is substituted by the SV 'the swimmers', which leads to the argument skeleton in (43). From the translation, we extract the translation of the pivot ('el padre de') and the location of the argument 'Imperial Savings & Loan' in target.

(43) 'the parent of the swimmers' → 'el padre de los nadadores'

Step 5

The last remaining chunk ('Imperial Savings & Loan') is judged to be ready for translation. It is embedded in a Static Context template mimicking the original context and sent to the baseline MT engine for translation. From the translation in (44), we extract 'Savings & Loan imperial' as the translation of the last chunk.

(44) 'The house of Imperial Savings & Loan' → 'La casa de Savings & Loan imperial.'

Step 6

After all chunks have been translated, the algorithm, in a final step, composes the output by stitching together the obtained translations in the target locations found by the SV translations. This leads to the final output in (45):

(45) 'Imperial Corp , *ubicado* en San Diego, *es* el padre de Savings & Loan imperial '

The translation in (45) improves on the original translation of the baseline MT system in (39). The main reason for the improvement is the fact that the reduction in syntactic complexity forces the baseline MT system to conjugate the verb 'to be' ('es' instead of 'ser') and to improve its translation for 'based' from the erroneous 'Fundar' to the correct 'ubicado'.

4.3 Substitution Variables

4.3.1 Introduction

In Section 4.2.4, we introduced the concept of Substitution Variables (SVs). SVs are replacement strings for the satellites in Figure 4.1 on page 41. They reduce the complexity of the original satellites, which can lead to an improved translation of the pivot. They are also used to track the location of the translation of the satellites in target.

In this section, we describe SVs more in depth. We discuss two different types of SVs (*Static SVs* and *Dynamic SVs*) and describe how their translation can be retrieved. We describe an experiment conducted to determine the optimal Static SV for nominal chunks, and discuss the results.

4.3.2 Early vs. Late MT Access

There are two ways to determine the translation of a Substitution Variable SV_{SAT_i} .

1. Early MT access: M translations $\{SV_{SAT_i}^1, \dots, SV_{SAT_i}^M\}$ with $M \leq Z$ are found before a TransBooster cycle. (Z is the number of all possible translations of SV_{SAT_i} by the baseline MT system).

2. Late MT access: a possible translation $\{SV_{SAT_i}^j\}$ with $1 \leq j \leq Z$ is determined *during* a TransBooster cycle, at run-time.

Since the baseline MT system is treated as a black box, it is not possible to determine all Z possible translations that the baseline system could generate for SV_{SAT_i} . It is possible, however, to find the M most likely translations by having SV_{SAT_i} translated in a number of straightforward contexts. For example, a baseline MT system might generate 3 different translations for the SV ‘the boy’: ‘el chico’, ‘el muchacho’ and ‘el niño’. In addition, in the case where this SV occurs in direct object position, the MT system will conflate the obligatory preposition ‘a’ with the previously found translations in Spanish.³ Therefore, although it is not feasible to determine with absolute certainty *all* Z possible translations of the Substitution Variable ‘the boy’, in this case we compose a list of $M = 6$ likely candidates $\{‘el chico’, ‘al chico’, ‘el muchacho’, ‘al muchacho’, ‘el niño’, ‘al niño’\}$ before a TransBooster run.

In the case of early MT access, we try to match each of the M candidate translations $SV_{SAT_i}^j$ ($1 \leq j \leq M \leq Z$) of each of the substitution variables SV_{SAT_i} against the string in (24). In the case of late MT access, we try to match the only candidate translation $SV_{SAT_i}^j$ ($1 \leq j \leq Z$) of each of the substitution variables SV_{SAT_i} against the string in (24) on page 44:

$$(27) [SV'_{SAT_1}] \dots [SV'_{SAT_i}] \textit{pivot}' [SV'_{SAT_{i+1}}] \dots [SV'_{SAT_{i+r}}]$$

In the latter case, $SV_{SAT_i}^j$ is the translation by the baseline MT system of SV_{SAT_i} in isolation, obtained during a TransBooster cycle.

4.3.3 Static vs. Dynamic Substitution Variables

The optimal SV to replace SAT_i is a string which reduces the complexity of SAT_i but shares its essential syntactic and lexico-semantic characteristics. An SV that does not reduce the complexity of the original sentence enough will be less likely to lead to an improvement of the translation of the pivot. On the other hand, a reduction in complexity can only help to improve the translation quality if essential syntactic and semantic similarity with the original constituent is maintained; an SV that differs too much from the

³This is a basic Spanish grammar rule: ‘I see the boy = Veo al(‘a + el’) chico/muchacho/niño.’

original could lead the analysis modules of rule-based baseline MT system astray, which might give rise to a distorted translation.

Therefore, the first obvious candidate to replace SAT_i is the string obtained by reducing SAT_i to its head, optionally accompanied by a determiner. We will refer to this type of substitution variable as a *Dynamic Substitution Variable* (DSV). For example, the DSV for the constituent ‘the chairman, a long-time rival of Bill Gates’ is ‘the chairman’. Since DSVs can only be obtained during the execution of the algorithm, the translation of these placeholders can only be obtained through late MT access.

Apart from the use of DSVs, it is equally possible to substitute the satellites with a predefined string, the translation of which can be determined by early MT access, before the execution of the TransBooster program. Unlike DSVs, which depend on the lexical content of the constituent they substitute for, these strings are predefined and can replace an entire class of constituents. We will refer to them as a *Static Substitution Variables* (SSVs). For example, an SSV for the constituent ‘the chairman, a long-time rival of Bill Gates’ could be ‘the man’. Unlike in the case of DSVs, there does not exist a one-to-one mapping between an SSV and the constituent it substitutes for. In other words, multiple suitable SSVs might be considered for the same constituent.

There exists a trade-off between accuracy and retrievability in the choice between SSVs and DSVs. SSVs, in principle, are easy to track in target since their possible translations can be determined before the actual execution of the algorithm (early MT access). However, they might distort the translation of the skeleton due to a lack of syntactic or semantic similarity with the argument they substitute for. DSVs, on the contrary, are expected to lead to a more accurate translation of the skeleton but are harder to locate in target since their translation has to be determined at run-time (late MT access).

4.3.4 Effects of SSV Schemata on Translation Quality

The experiment outlined in this section was performed at the very start of the TransBooster project. Its objective was to measure the quality of 5 different SSV schemata for the TransBooster approach of satellite replacement. The two main questions we wanted to address are the following:

1. Is it possible to rely solely on SSVs for safe satellite substitution?
2. What are the best SSVs for each of the baseline MT systems involved?

4.3.4.1 SSVs

We experimented with five different SSV schemata, ranging from non-word strings to placeholders syntactically similar to the original constituents. In the experiment we focused on the replacement of NP arguments in a verbal context. Table 4.1 contains a description of each SSV schema and illustrates its use by substituting the arguments ‘The man, a long-time rival of Bill Gates’ and ‘fast and confidential deals’ in example sentence (20) on page 40.

SSV schema	Description / Example
Non-word strings	Strings not present in the lexicon of the baseline MT system, no syntactic/semantic resemblance to original. e.g. ‘ <i>SUBJ1 likes OBJ1</i> ’
Non-word strings with determiner	Non-word strings preceded by determiner. e.g. ‘ <i>The SUBJ1 likes the OBJ1.</i> ’
Acronyms	Sometimes present in the lexicon of the baseline MT engine, no syntactic/semantic resemblance to original e.g. ‘ <i>IBM likes CD.</i> ’
Proper nouns	Sometimes present in the lexicon of the baseline MT engine, no syntactic/semantic resemblance to original. e.g. ‘ <i>Mary likes John.</i> ’
Controlled heads	Always present in the lexicon of the baseline MT engine, syntactic resemblance to original. e.g. ‘ <i>The man likes the woman.</i> ’

Table 4.1: Substitution Variables for NP-type constituents

The SSVs in Table 4.1 are ranked from simple non-word strings to more complex controlled heads. Non-word strings, with or without determiners, are not present in the dictionaries of baseline rule-based MT systems and are therefore treated as unknown words. Since they are usually left untranslated, they are very easy to track in target. Like non-word strings, acronyms and proper nouns do not bear any semantic similarity to the constituent they substitute, but they might be present in the baseline MT lexicon. Therefore they are more likely to be correctly analysed by the MT’s analysis module. This increases the probability of a correct translation of the pivot. The translation of both acronyms and proper nouns by the baseline MT system can be easily deduced by

early MT access. Finally, controlled heads are SVs that mimic the syntactic structure of the constituent they substitute for. Of all SSVs, they are the ones that bear the closest syntactic resemblance to the original constituents and therefore are, in theory, the SSVs less likely to distort the translation of the pivot. As in the case of acronyms and proper nouns, their translation is obtained by early MT access.

4.3.4.2 Experimental Setup

In order to test the effect of the SSV schemata in Table 4.1 on the translation of the pivot and the location of the translation of the satellites in target, we constructed a corpus of test sentences based on the most frequent verbal subcategorisation frames in the Penn-II Treebank. A subcategorisation frame specifies the arguments that a predicate must take in order to form a complete grammatical construction. The subcategorisation frames we used were extracted automatically (O'Donovan, 2006) from a version of the Penn-II Treebank enhanced with LFG (Lexical Functional Grammar) f-structure information (Burke, 2006).

Summarised very briefly, LFG is a unification-based grammar introduced by Kaplan and Bresnan (1982) that minimally contains two levels of representation: c(onstituent)-structure and f(unctional)-structure. C-structure represents language-specific syntactic surface information in the form of CFG trees. F-structure uses recursive attribute-value feature structures to encode abstract syntactic information about predicate-argument-modifier relations and certain morphosyntactic properties such as tense, aspect and case. O'Donovan (2006) used the version of the Penn-II treebank which had previously been enhanced by Burke (2006) with 'functional annotations'⁴ to automatically derive subcategorisation frames for all predicates in the Treebank. For example, the subcategorisation frame of the predicate 'use' in the sentence 'He uses an example to illustrate the concept' is shown in (46):

(46) use([subj,obj,xcomp])

Table 4.2 contains the most important syntactic functions that can occur in LFG f-structures. As we will further explain below, we used the most frequent verbal subcategorisation frames thus derived to construct a corpus of test sentences for the experiment.

⁴Linking information between c and f-structures that is present on the c-structure nodes.

The subcategorisable grammatical functions that can occur in a LFG semantic form are listed in Table 4.2 together with a brief description

Grammatical Function	Description
SUBJ	Subject
OBJ	Direct Object
OBJ2	Indirect Object
OBL	Oblique Argument
COMP	Closed Verbal Complement (containing own subject)
XCOMP	Open Verbal Complement (not containing own subject)
PART	Particle
POSS	Possessive

Table 4.2: Subcategorisable syntactic functions in LFG.

We reduced the 577 different verbal subcategorisation frame types occurring in the Penn-II treebank to 38 frame types by conflating all prepositions and particles. From the resulting 38 frame types, we extracted the 10 most frequent types. Subcategorisation frames containing only subjects were ignored, as they provided the least room for simplification. Table 4.3 contains the 10 most frequent subcategorisation frames.

Subcat. frame	Voice	Occurrences
subj_obj	active	39881
subj_xcomp	active	14577
subj_obl	active	8234
subj_obj_obl	active	7092
subj_comp	active	5796
subj_obl	passive	3062
subj_xcomp	passive	2049
subj_obj_xcomp	active	1697
subj_part_obj	active	1674
subj_obj_comp	active	458

Table 4.3: The 10 most frequent verbal subcategorisation frames in the Penn Treebank, in descending frequency and excluding subcategorisation frames containing only subjects

For each of the subcat frame types in Table 4.3, verb lemmas corresponding to the frame were extracted from the treebank. For each frame-lemma pair, two sets of 6 sentences were constructed: one with the predicate in the simple past, the other with the predicate in the future. We chose to generate verb forms in the simple past and future tense to minimise the possibility of noun-verb misanalyses by the baseline MT engines. The sentences in

(47) and (48), translated from English→Spanish by Systran, are examples in which verbs in the simple present are misanalysed as nouns, making the output unintelligible, whereas the simple past and future tense give acceptable results.

- (47) a. 'The rider spurs the horse.' → *'Los estímulos del jinete el caballo '(literal backtranslation = 'The stimuli of the rider the horse ')
- b. 'The rider will spur the horse.' → 'El jinete estimulará el caballo '
- (48) a. 'The explanation prods the student to think ' → *'Los golpecitos de la explicación el estudiante a pensar.'(literal backtranslation = 'The punches of the explanation the student to think.')
- b. 'The explanation prodded the student to think.' → 'La explicación pinchó a estudiante para pensar.'

Each set contained a reference sentence with dummy arguments and 5 test sentences in which the argument slots were replaced by one of the 5 different SSV schemata in Table 4.1, as is shown in Table 4.4.

1	Reference	$[ARG_1] \dots [ARG_l]$	pivot	$[ARG_{l+1}] \dots [ARG_{l+r}]$
2-6	SSV substitutions	$[SV_{ARG_1}^1] \dots [SV_{ARG_l}^2]$	pivot	$[SV_{ARG_{l+1}}^1] \dots [SV_{ARG_{l+r}}^1]$

Table 4.4: A test set containing a reference sentence and 5 test sentences for a particular frame-lemma pair l = number of arguments to left of pivot, r = number of arguments to right of pivot, $1 \leq i \leq 5$.

For example, for the frame-lemma pair *include*([*subj,obj*]), two sets of 6 sentences were constructed, one in the simple past, the other in the future tense Table 4.5 contains one of those sets the reference sentence 'The woman included the man' and 5 test sentences in the simple past obtained after replacing the arguments of the original predicate by the SSV schemata.

	SSV schema	Generated sentence		
1	Reference	The woman	included	the man.
2	Non-word strings	SUBJ1	included	OBJ1.
3	Non-word strings with det	The SUBJ1	included	the OBJ1.
4	Acronyms	IBM	included	CD.
5	Proper nouns	Mary	included	John.
6	Controlled heads	The cat	included	the skunk.

Table 4.5: A test set containing a reference sentence and 5 test sentences for the frame-lemma pair *include*([*subj,obj*]).

The two main goals of the experiment were:

1. to compare the translation of the pivot in the test sentences to the translation of the pivot in the reference sentence.
2. to compare the position of the translation of the SSVs (SV_{ARG_j}) in the test sentences against the position of the translation of the original arguments (ARG_j) ($1 \leq j \leq l+r$).

Table 4.6 contains the number of sentences selected per frame, as determined by verb lemmas attested for the frame in the Penn-II Treebank. The verb forms were extracted directly from the Penn-II treebank, which explains the different numbers for sentences in past and future tense for the same frame.

Subcat. Frame	Voice	Tense	Nr. extracted Sentences
subj_obj_comp	active	future	96
subj_comp	active	future	292
subj_obj_obl	active	future	1347
subj_obj_part	active	future	403
subj_obj	active	future	1613
subj_obj_xcomp	active	future	325
subj_obl	active	future	1280
subj_xcomp	active	future	392
subj_obj_comp	active	past	93
subj_comp	active	past	280
subj_obj_obl	active	past	1252
subj_obj_part	active	past	376
subj_obj	active	past	1271
subj_obj_xcomp	active	past	303
subj_obl	active	past	1244
subj_xcomp	active	past	401
subj_obl	passive	future	863
subj_xcomp	passive	future	212
subj_obl	passive	past	863
subj_xcomp	passive	past	212
Total			13118

Table 4.6: Counts for the 10 most frequent subcategorisation frames.

By filling the arguments slots in the 13118 templates (cf. Table 4.6) with one dummy variable and the 5 SSV schemata, we obtained a test corpus of 78,708 sentences. These sentences were translated from English into Spanish by the 4 baseline RBMT systems that we introduced in Chapter 3: Systran, LogoMedia, PromT and SDL. Since we did

not possess in-house versions of the above mentioned MT systems at the time of the experiment, we had to rely on their free on-line versions, which put a size restriction on the input files. We therefore decided to split the test corpus into a number of smaller files, with a maximum size of 64Kb each. These files were uploaded onto a web server and translated by executing a script that retrieves the MT output of the test files by using WGET. Translating the test corpus of 78,708 sentences by 4 MT engines resulted in a total of 314,832 translated sentences to be evaluated.

The translation of a test sentence was deemed successful if the following two conditions were satisfied:

1. The translation of the pivot in the test sentence is identical to the translation of the pivot in the reference sentence.
2. The translated SSVs ($SV_{ARG_j}^i$) are in the same position with respect to the pivot as the translated original arguments (ARG_j)

For each of the four MT systems, a list of possible translations of the SSVs was obtained (early MT access). We then used a string comparison script to automatically check the 314,832 translations obtained for the quality of the pivot and for correctness of the location of the arguments in the target language.

4.3.4.3 Results

Tables 4.7 to 4.10 contain the results of successful SSV replacement for LogoMedia, Systran, SDL and PromT respectively. The first column (*worst frame*) in each table contains the success rate of the SSV replacement for the worst performing subcategorisation frame. For example, the worst frame success rate for the 'proper noun' SSV in Table 4.7 is 75.31%. This means that substituting the arguments with 'proper noun' SSVs leads to 75.31% successful sentences for the worst frame of the 20 different subcategorisation frames in Table 4.6. The second column (*best frame*) contains the success rate of the SSV replacement for the best performing subcategorisation frame. The third column (*average*) contains the weighted average of the SSV replacement over all 20 subcategorisation frames, where the weight of a subcategorisation frame equals the number of sentences selected per frame in

the Penn-II Treebank, or $average = \frac{\sum_{i=1}^{20} w_i x_i}{\sum_{i=1}^{20} x_i}$ with x_i the success rate for subcategorisation frame i and w_i the number of sentences selected for frame i . For example, the average success rate for the ‘proper noun’ SSV in Table 4.7 is 95.26%. This means that substituting the arguments with ‘proper noun’ SSVs leads, on average, to 95.26% successful sentences by taking a weighted average over all 20 different subcategorisation frames in Table 4.6

The first row in each table (*optimal combination*) contains the success rate of the best SSV replacement per subcategorisation frame, i.e. the replacement by the SSV candidate that achieved the highest score for the *individual* subcategorisation frame in question. For example, the average success rate for the ‘optimal combination’ in Table 4.7 is 95.50%. This means that substituting the arguments with the best possible SSV schema per frame leads, on average, to 95.50% successful sentences by taking a weighted average over all 20 different subcategorisation frames in Table 4.6.

The subsequent rows contain the scores for the argument replacement of *all* subcat frames by the same SSV.

SSV	worst frame success (%)	best frame success (%)	average success (%)
Optimal combination	75.31	100.00	95.50
Proper nouns	75.31	100.00	95.26
Non-word strings with det	5.56	90.71	71.12
Non-word strings	4.29	92.50	69.69
Controlled heads	5.45	90.71	70.50
Acronyms	5.56	88.21	66.75

Table 4.7: Results of SSV replacement on translation quality for LogoMedia

SSV	worst frame success (%)	best frame success (%)	average success (%)
Optimal combination	86.09	100.00	97.22
Proper nouns	54.29	100.00	93.34
Controlled heads	4.06	100.00	81.03
Non-word strings with det	3.77	99.66	79.24
Acronyms	10.85	99.32	76.35
Non-word strings	4.06	98.97	73.03

Table 4.8: Results of SSV replacement on translation quality for Systran

SSV	worst frame success (%)	best frame success (%)	average success (%)
Optimal combination	4.25	100.00	84.12
Non-word strings with det	2.83	100.00	83.88
Controlled heads	2.83	100.00	83.68
Non-word strings	2.83	100.00	82.14
Proper nouns	4.25	100.00	82.02
Acronyms	2.83	100.00	81.41

Table 4.9: Results of SSV replacement on translation quality for SDL

SSV	worst frame success (%)	best frame success (%)	average success (%)
Optimal combination	97.34	100.00	99.16
Proper nouns	97.21	100.00	98.70
Acronyms	40.40	99.74	92.24
Controlled heads	6.03	99.66	79.76
Non-word strings with det	4.98	99.75	78.44
Non-word strings	4.87	99.32	76.87

Table 4.10: Results of SSV replacement on translation quality for PromT

4.3.4.4 Analysis

Two different SSV replacement strategies might be considered:

1. Best overall SSV replacement. The replacement schema shown to work best over the totality of the test corpus for a particular MT engine is applied to all sentences, irrespective of the subcategorisation frame of its verb.
2. Best individual SSV replacement. The replacement schema applied to a sentence is the one shown to work best for the particular subcategorisation frame of the predicate of that sentence

For LogoMedia, Systran and PromT, the best overall SSV replacement scores are achieved by the proper noun SSV schema, with average scores of 95.26%, 93.34% and 98.70% respectively. This result can be explained by the fact that the chosen proper noun SSVs are semantically more similar to the constituent they substitute for than the other SSV candidates. For instance, in Table 4.5 the SSV 'Mary' resembles the original constituent 'The woman' more than other SSVs such as 'SUBJ1', 'IBM' or 'The cat'. Substituting arguments with semantically different SSVs can easily lead to a distortion in the translation of the pivot, as is shown in Tables 4.11 and 4.12.

Source	Target
The woman strapped the man	La mujer azotó al hombre
The SUBJ1 strapped the OBJ1	El SUBJ1 ató con correa el OBJ1
SUBJ1 strapped OBJ1	SUBJ1 OBJ1 corto de dinero
IBM strapped CD	IBM ató con correa CD
Mary strapped John	Mary azotó a John
The cat strapped the skunk	El gato ató con correa a la mofeta

Table 4.11: Translation of the test set for the frame-lemma pair *strap*(*[obj, subj]*) by Logomedia

In Table 4.11, the pivot is translated as 'azotar' (= 'to whip') for the 'proper noun' SSV. In the case of acronyms, controlled heads and non-word strings with determiner, the pivot is translated as the idiom 'atar con correa' (= 'to put on a lead'). The use of non-word strings without determiners leads to an erroneous homograph resolution of 'strapped', which is translated as an adjective ('corto de dinero' = 'not rich').

Source	Target
The woman will face the man	La mujer se encontrará cara a cara con el hombre
The SUBJ1 will face the OBJ1	El SUBJ1 mirará hacia el OBJ1
SUBJ1 will face OBJ1	SUBJ1 mirará hacia OBJ1
IBM will face CD	IBM enfrentará CD
Mary will face John	Mary se encontrará cara a cara con John
The cat will face the skunk	El gato mirará hacia la mofeta

Table 4.12: Translation of the test set for the frame-lemma pair *face*(*[obj, subj]*) by Logomedia

In Table 4.12, the pivot is translated as 'encontrarse cara a cara' (= 'to be face to face') for the proper noun SSV. In the case of controlled heads and non-word strings with and without determiner, the pivot is translated as 'mirar hacia' (= 'to look to'). The use of acronyms leads to a pivot translation of 'enfrentar' (= 'to confront').

For SDL, the best overall SSV replacement score for is achieved by the 'non-word string with determiner' SSV schema, with an average score of 83.88%. The variation between the scores of the different SSVs, however, is less pronounced for SDL (81.41%-83.88%) than for the other MT systems (cf. Tables 4.7 - 4.10). This can be explained by the fact that the SDL engine is probably less context-sensitive than the others, i.e. it relies less on the semantic properties of the arguments than the other MT systems to select a translation of the pivot.

Also, the overall results for SDL were significantly lower than for the other MT engines. This is caused by the fact that SDL often relies on the mediopassive voice in Spanish, a grammatical construction which subsumes the meanings of both the middle voice and the passive voice. Spanish, apart from the traditional periphrastic passive construction, can express the passive and middle voice with a synthetic construction in which the syntactically active verb, accompanied by a reflexive pronoun, has a semantically passive character.⁵

The average scores for the best individual SSV replacement (*'optimal combination'* in Tables 4.7 to 4.10) range between 84.12% for SDL to 99.16% for PromT. However, even this optimal selection obtains rather low scores for certain frames: the scores for the worst performing frames vary between 4.25% for SDL to 97.34% for PromT.

Taking into account the simplified nature of the sentences and the fact that we extracted verb forms in the simple past and future tense to minimise the possibility of verb-noun ambiguities, it is to be expected that these scores would be lower in a real-world environment. Therefore, it is not true that constituents can be safely replaced with SSVs for any frame. A reasonable implementation of the replacement algorithm would involve a replacement schema in which the SVs maintain the maximum syntactic and lexico-semantic similarity with the original constituent they substitute for.

4.3.5 Conclusion

The experiment in Section 4.3.4 shows that it is not possible to rely solely on SSVs for a safe satellite substitution. We will, therefore, opt for a backoff schema in which we first attempt to substitute satellite chunks with DSVs, the translation of which is determined by late MT access, and fall back on SSVs in case the DSV substitution is not successful.

4.4 Summary

This chapter introduces the concepts necessary to understand the technical details of TransBooster, which are presented in Chapter 5. In the first part of this chapter, we explained the basics of the decomposition algorithm and illustrated its working with several

⁵E.g. 'El piso es vendido' (periphrastic) vs 'El piso se vende' (synthetic)

examples. In the second part, we expanded on the concept of Substitution Variables and reported the results of a preliminary experiment conducted to determine the suitability of various Static Substitution Variable schemata.

Chapter 5

TransBooster Architecture: Technical Details

5.1 Introduction

In this chapter, we expand on the concepts introduced in Chapter 4 and treat the different components of the TransBooster architecture in detail. In the first part (Section 5.2: TransBooster Mark I), we explain the standard TransBooster algorithm. The second part (Section 5.3: TransBooster Mark II) contains an outline of an alternative, simplified TransBooster strategy.

5.2 TransBooster Mark I

This part is structured as follows: we first focus on head identification (Section 5.2.1), the construction of pivots (Section 5.2.2) and the distinction between arguments and adjuncts (Section 5.2.3). We provide an in-depth description of how the Substitution Variables introduced in the previous chapter are constructed (Section 5.2.4) and explain how context templates are constructed and used (Section 5.2.5). We then examine the back-end of the TransBooster engine (Section 5.2.6) and present the safety measures that have been put in place to prevent erroneous decomposition (Section 5.2.7). Finally, we provide a summary of the algorithm (Section 5.2.8) and illustrate its working with an example.

5.2.1 Identifying Heads

As outlined in Section 4.2.2 on page 42, the first step in determining the pivot of a constituent is the identification of its head. We use the head-finding rules of (Cahill, 2004), which are an adaptation of the head-lexicalised grammar annotation scheme of (Magerman, 1995) and (Collins, 1999). The rules are displayed in Table 5.1. The first column contains the constituents of which we want to determine the head. The second column indicates the direction in which the children of the constituent will be scanned. The third column contains a list of candidate head categories for each constituent.

The head-finding function proceeds as follows: for each candidate head category X in the third column, starting with the first category, scan the children of the constituent from left to right (head-initial constituents) or right to left (head-final constituents). The first child that matches category X is the head node. If no child matches any category in the list, the first child, in the case of head-initial constituents, or the last child, in the case of head-final constituents, is considered to be the head.

Asterisks (****) indicate the beginning of a list of categories that, if possible, should not be chosen as the head of the constituent. If a child is found whose category differs from those occurring after the asterisks, that child is considered to be the head. If all children match one of the categories after the asterisks, choose the leftmost or rightmost child according to the search direction. For categories without any values (-), choose the leftmost or rightmost child according to the search direction.

The head of an NP node is determined by a separate set of rules. The first child encountered whose category label begins with N in a right-to-left scan is the head, if the following two conditions are met: (i) the category label does not contain a Penn-II functional tag, and (ii) if the category label is NP, it must not be preceded by punctuation. If no category is found, the algorithm relies on the information in Table 5.1 to determine the head of the NP.

In the case of a coordinated node N^1 , the default head finding procedure as explained

¹Note that the constituent CONJP in Table 5.1 refers to multi-word conjunctions dominating a limited amount of lexical items (e.g. 'as well as', 'rather than', 'not to mention' etc). CONJP constituents are never subject to recursion. In the Penn Treebank, coordinated phrases of the same syntactic category X are joined under a mother node X . Coordinated phrases of a different syntactic category are joined under the mother node UCP ('Unlike Coordinated Phrase')

in this section is overridden. If two phrasal constituents of the same category are coordinated, the first CC-labelled constituent (the coordination word) found while scanning the children from left to right is assigned to be the head. During the tree-flattening procedure, the CC node is analysed as the pivot of the chunk and the two coordinating constituents are analysed as adjuncts, since they are not necessary for a safe translation of the CC. Both coordinated constituents are subject to recursive decomposition in a later stage. In all other cases ((i) N contains more than two coordinated constituents, (ii) N contains two coordinated phrasal constituents of a different category, or (iii) in addition to CC, N contains at least one lexical item), N is not decomposed further but sent as a single unit to the baseline MT system for translation.

Constituent	Direction	Candidates
ADJP	Right	% QP JJ VBN VBG ADJP \$ JJR JJS DT FW IN **** RBR RBS RB
ADVP	Left	RBR RB RBS FW ADVP CD **** JJR JJS JJ NP
CONJP	Left	CC RB IN
FRAG	Left	-
INTJ	Right	-
LST	Left	LS :
NAC	Right	NN NNS NNP NNPS NP NAC EX \$ CD QP PRP VBG JJ JJS JJR ADJP FW
NP	Right	EX \$ CD QP PRP VBG JJ JJS JJR ADJP DT FW RB SYM PRP\$ **** PRN POS
PP	Left	IN TO FW
PRN	Left	-
PRT	left	RP
QP	Right	\$ % CD NCD QP JJ JJR JJS DT
RRC	Left	VP NP ADVP ADJP PP
S	Right	TO VP SBAR ADJP UCP NP PP-PRD ADJP-PRD NP-PRD
SBAR	Right	IN S SQ SINV SBAR FRAG X
SBARQ	Right	SQ S SINV SBARQ FRAG X
SINV	Right	MD IN VBZ VBD VBP VB AUX VP S SINV ADJP NP
SQ	Right	MD VBZ VBD VBP VB AUX VP SQ
UCP	Left	CC S **** ADVP RB PRN
VP	Left	MD VBD VBN VBZ VB VBG VBP POS AUX AUXG VP TO ADJP JJ NP
WHADJP	Right	JJ ADJP
WHADVP	Left	WRB
WHNP	Right	NN NNS NNP NNPS NP WDT WP WP\$ WHADJP WHPP WHNP
WHPP	Left	IN TO FW
X	Left	-

Table 5.1: Tree Head Table – the list of head-finding rules based on (Magerman, 1995)

5.2.2 Constructing Pivots

We have provided an introduction to pivots in sections 4.2.1 and 4.2.2 on pages 40 and 42. The main goal of identifying a pivot for each input chunk is to obtain the part of the input string that has to remain intact during the decomposition process. We achieve this by extending the chunk’s nucleus (cf. Section 5.2.1) with necessary lexical information (adjacent terminal strings) to (i) capture possible idiomatic constructions, and (ii) avoid substituting arguments with a limited lexical scope.

An example² of an idiomatic construction is shown in (49):

- (49) ‘close to the border’→‘cerca de la frontera’.
‘close’→*‘ciérrese’.
‘to the border’→‘a la frontera’.

In (49), the translation of the head ‘close’ of the ADJP ‘close to the border’ in isolation leads to the deficient translation ‘ciérrese’ (= reflexive imperative of the verb ‘to close’). In order to obtain the correct translation ‘cerca de’, the preposition ‘to’ has to be adjacent to the head ‘close’ in the pivot string sent to the MT engine. This can be achieved in two different ways:

1. Include the necessary lexical material (e.g. ‘to’) in the pivot.
2. Include the necessary lexical material (e.g. ‘to’) in the argument skeleton.

In (49), the first option would lead to a decomposition of the input string into the pivot ‘close to’ and NP argument ‘the border’. The second option would lead to the pivot ‘close’ and PP argument ‘to the border’.³ Although both options lead to the correct translation ‘cerca de la frontera’, option 1 is preferable to option 2, because the amount of variations of possible translations for an NP SSV is usually less than for a PP SSV, due to the highly idiomatic character of prepositions. In other words, the total number of possible translations Z (cf. Section 4.3.2 on page 54) is usually higher for a PP than for

²All examples in this chapter were translated from English→Spanish by LogoMedia.

³The ADJP ‘close to the border’ does not contain enough lexical material to be eligible for decomposition in a real-world TransBooster scenario, as is further explained in Section 5.2.6. We included this short example here for purposes of clarity. It is easy to see, however, how the constituent could be extended with modifiers (e.g. ‘close to the dangerous border that separates Paraguay from Bolivia’) in which case it would be subjected to decomposition.

an NP since the translation of the head of the PP, the preposition, depends heavily on the preceding context. Therefore, it is more difficult, both in the case of early MT access as in the case of late MT access, to successfully determine the translation of the argument SSV in the case of a PP SSV than in the case of an NP SSV. Accordingly, even though it is not strictly necessary to include additional lexical material in the pivot, we find it preferable to extend the pivot with a limited amount of idiomatic lexical items than to have to account for a large number of SSV translations.

The second case in which we choose to extend the pivot with adjacent lexical information is to avoid having to substitute arguments with a limited lexical scope. If, in Figure 4.1 on page 41, an argument satellite containing only a few words is adjacent to the pivot, the SSV replacement of this satellite is unlikely to lead to a simplification of the input due to the syntactic simplicity of the original constituent. On the contrary, since the possibility of a deficient translation of the pivot due to semantic differences with the original can never be ruled out, an SSV replacement in these cases is likely to do more harm than good. Therefore, argument satellites adjacent to the head and dominating fewer leaf nodes than a predefined threshold N are included in the pivot. The optimal value of the threshold N depends on the baseline MT system used and was empirically established by tuning the program parameter `p_PivotAttach`, as will be further explained during the discussion of experimental results in Chapter 6. For example, the best results for baseline MT system LogoMedia were achieved with $N = 2$. As an example, consider the sentence in (50):

- (50) ‘[Traders]_{ARG₁} [said]_{pivot} [most of their major institutional investors, on the other hand, sat tight]_{ARG₂}’. → ‘[Traders said]_{pivot} [most of their major institutional investors, on the other hand, sat tight]_{ARG₂}’.

Substituting the first argument ‘Traders’ with an SV would not improve the syntactic complexity of the sentence. It could only lead to a possible distortion of the translation of the pivot ‘said’. Therefore, it is included in the pivot.

Like satellite chunks, the pivot is a translatable chunk, provided it is embedded in a sufficient context. However, contrary to satellites, which are embedded in a context template, the context for the pivot is provided by SVs, thereby simplifying the original arguments.

The identification of the pivot makes it possible to flatten the original Penn-II-style tree into a simpler structure consisting only of the pivot and satellites (cf. Figure 4.3 on page 41). In this simpler structure, the pivot is the point of reference with respect to which we will calculate the position of the translation of the satellites in target. Given a Penn-II-style tree, we use two different strategies to construct the pivot. In a first step, the syntactic characteristics of the constituent are matched against one of the patterns designed to take into account idiomatic constructions and arguments with a limited lexical scope (cf. Section 5.2.2.2). If no match is found, in a second step a default generic pivot finding procedure is used (cf. Section 5.2.2.1).

5.2.2.1 Constructing Pivots: Default Tree Flattening

The default pivot finding procedure is independent of the syntactic properties of the input chunk. It only takes into account the lexical coverage of the input chunk's head nodes along the tree's head projection line. The procedure starts at the root node of the input chunk, recursively traverses the tree and, at each step, examines the local head node N . If N dominates $\leq L$ leaf nodes, the node N and its yield is taken to be the pivot. If, on the other hand, the head node N contains too many leaf nodes ($> L$), the procedure considers the head node N' immediately dominated by N along the constituent's head projection line, to be a pivot candidate, and so on, until a head with L words or less in its coverage is found. L is a parameter that allows us to experiment with varying maximum pivot lengths. The optimal value of L depends on the baseline MT system used and was determined empirically by tuning the program parameter `p_PivotLength`.⁴

As an example, consider Figure 5.1. The pivot finding procedure starts at node 1, the node representing the input chunk 'A ... L'. Node 3, the head of node 1, dominates 11 lexical items (B ... L). Since this number is greater than the threshold $L = 4$, the procedure examines node 4, the head of node 3. Node 4 dominates only 1 ($\leq L$) lexical item, namely 'B'. Therefore 'B' is the pivot of 'A ... L'. Nodes 2 and 5 are the pivot's satellites. The resulting flattened tree structure will serve as input to TransBooster's

⁴The best BLEU, NIST and GTM scores were achieved with $L = 4$ for all tree baseline MT systems. Cf. Section 6.2.2 on page 102.

decomposition module.⁵

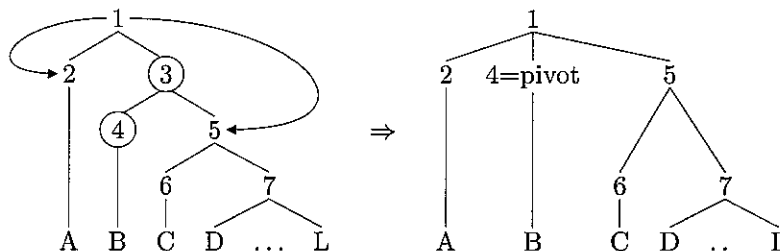


Figure 5.1 Basic tree flattening. 1-7 are arbitrary non-terminal categories. A-L are lexical items. Node 3 is the head of node 1. Node 4 is the head of node 3. The resulting flattened tree on the right-hand side is the input to TransBooster’s decomposition module.

5.2.2.2 Constructing pivots: Extended Tree Flattening

Contrary to the default tree flattening procedure, the extended tree flattening strategy takes into account the syntactic characteristics of the input chunks. It tries to match these characteristics against a number of previously determined syntactic templates. The goal of the templates is to cover most of the cases in which it is preferable to extend the pivot with additional lexical material to account for idiomatic constructions and specific syntactic phenomena, as explained at the start of Section 5.2.2. If none of the templates matches the syntactic environment of the input chunk, its tree is flattened by using the default tree flattening procedure.

As an example, consider Figure 5.2. The left-hand side tree is identical to the left-hand side tree in Figure 5.1. In this case, the specific syntactic configuration in the tree matches a predefined template, which for example expands the pivot with its first adjacent non-empty node to its right. As a consequence, ‘C’, the lexical coverage of node 6, is attached to the pivot ‘B’, leading to pivot ‘BC’ and a different tree flattening.

The pivot templates were constructed based on a manual analysis of the most frequent occurrences of non-terminal expansions in the training section of the Penn-II Treebank. For this analysis, we relied on the treebank grammar used by (Cahill, 2004), which was constructed following (Charniak, 1996). A treebank grammar is a context-free grammar (CFG) created by reading off the production rules directly from hand-parsed sentences

⁵Note that pivot finding and tree flattening are recursively applied to satellites (here nodes 2 and 5)

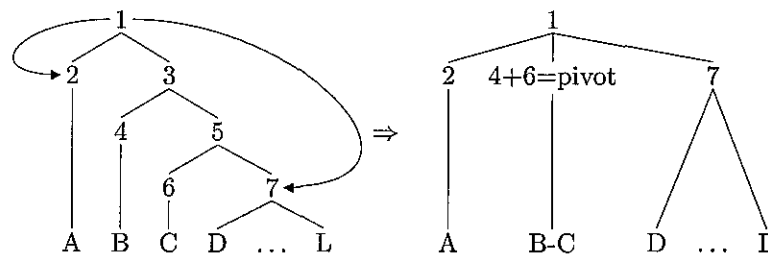


Figure 5.2 Extended tree flattening 1–7 are arbitrary non-terminal categories. A–L are lexical items. Node 3 is the head of node 1. Node 4 is the head of node 3.

in a treebank. In the case of (Cahill, 2004), the CFG was constructed based on the training section of the Penn-II treebank (sections 01–22), with empty productions and trace information removed and all Penn-II functional information tags attached to CFG categories stripped

Since it is not possible to manually analyse all 17,034 rule types in the treebank-based CFG, we chose to investigate the most frequent rules that account for 85% of rule tokens per non-terminal. Given that it is not useful to subject chunks with a limited lexical range to decomposition, we excluded the rules dominating an average of fewer than 4 leaf nodes. This figure is related to the optimal value of the parameter `p_ChunkLength`, introduced on page 50⁶ After these reductions, 554 rule types remained for analysis. The rules were analysed by examining the corresponding rule-token dominated sub-trees in the treebank. Two different tools were found useful for this analysis: TGREP (Pito, 1993) and the DCU Treebank Tool Suite (TTS)⁷ (Cahill and van Genabith, 2002). TGREP is a well-known Unix-based tool that allows parse-annotated tree searches in the same spirit as GREP. TTS is a web-based treebank inspection tool developed at DCU with extended functionality for PCFG parsing. TGREP supports searches of arbitrary tree fragments and depth, whereas TTS is easy to use and displays the results graphically.

After inspecting individual instances of each relevant rule type, we derived specific coding guidelines Appendix B contains a list of the main extended pivot treatment procedures for non-terminal nodes in the Penn-II Treebank. Each rule is illustrated with an example.

⁶All TransBooster program parameters are summarised in Section 6.2.2

⁷<http://www.computing.dcu.ie/~acahill/tts/>

5.2.3 Arguments vs. Adjuncts

As pointed out in Section 4.2.3 on page 43, we broaden the traditional notion of the term ‘argument’ to those nodes that are essential for the correct translation of the parent node. Nodes labeled as adjuncts can be safely omitted in the string sent to the baseline MT system to obtain the translation of the pivot. Omitting redundant material in the original string is a first obvious way to simplify the input. However, caution must be taken not to omit certain lexical items that, despite being classified as adjuncts in a certain grammatical framework, are nevertheless essential to guarantee a correct translation of the pivot.

For example, consider the Penn-II sentence in Figure 5.3. If, in Figure 5.3, the directional ADVP ‘down’ and PP ‘into their canal’ are labelled as adjuncts, the translation of the resulting argument skeleton ‘We were coming’ would lead to the erroneous pivot translation *‘viniendo’ (‘coming from somewhere’) instead of the correct ‘bajando’ (‘coming/going down’), as is represented in (51).

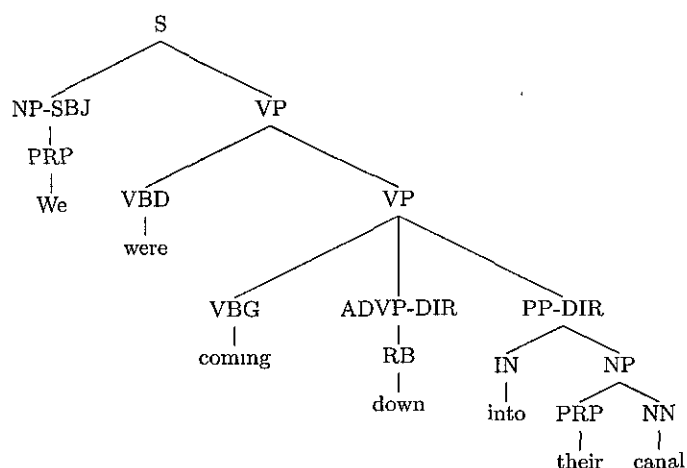


Figure 5.3: Penn-II tree representation of ‘we were coming down into their canal.’

- (51) ‘We were coming down into their canal.’ → ‘Estabamos bajando en su canal’.
 ‘We were coming’ → ‘Estabamos viniendo’.

Likewise, consider the Penn-II sentence in Figure 5.4:

If, in Figure 5.4, the directional ADVP ‘away from the stock market’ is labelled as an adjunct, the translation of the resulting argument skeleton ‘Individual investors have

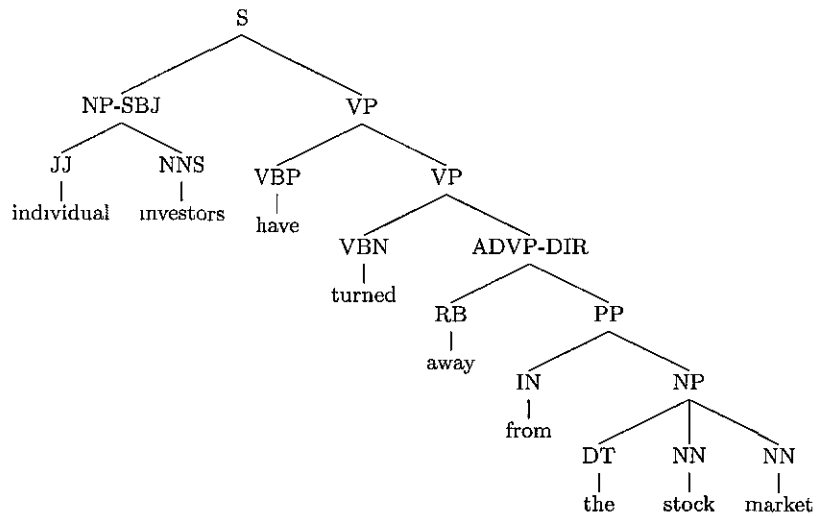


Figure 5.4: Penn-II tree representation of ‘Individual investors have turned away from the stock market’

turned’ would lead to the erroneous pivot translation *‘han doblado’ (‘have turned’) instead of the correct ‘se han alejado’ (‘have turned away’), as is shown in (52):

- (52) ‘Individual investors have turned away from the stock market’ → ‘Los inversores particulares se han alejado del mercado de valores’
 ‘Individual investors have turned’ → ‘Los inversores particulares han doblado’.

Therefore, the argument-adjunct distinction of a specific grammatical framework can only serve as a basis to distinguish between ‘pivot arguments’, essential nodes for the correct translation of the pivot, and ‘pivot adjuncts’ (redundant material) among satellites. A thorough investigation of the most frequent phenomena is necessary to avoid errors as shown in (51) and (52).

In the first phase of determining argument-adjunct distinction guidelines for node-labelling by TransBooster, we relied on information provided in (Hockenmaier, 2003). In this work, the author presents the creation of training data and the development of probability models for statistical parsing of English with Combinatory Categorical Grammar (CCG, (Steedman, 1996, 2000)). CCG is a lexicalist, constraint-based grammatical theory in which categories are the building blocks of the grammar. Words are associated with very specific categories which define their syntactic behaviour. In order to obtain training data for CCG parsing, Hockenmaier (2003) had to transform the phrase-structure

trees in the Penn Treebank into a corpus of CCG derivations. In the Penn Treebank, the complement-adjunct distinction is not marked explicitly, as is clearly stated in (Marcus et al , 1994):

“After many attempts to find a reliable test to distinguish between arguments and adjuncts, we have abandoned structurally marking this difference. Instead, we now label a small set of clearly distinguishable roles, building upon syntactic distinctions only when the semantic intuitions are clear cut.”

In the implementational details of the transformation from Penn Treebank to CCGbank in (Hockenmaier, 2003), however, clear guidelines are provided to distinguish arguments from adjuncts, based on heuristic procedures which rely on the label of a node and its parent.

In the second phase of our argument-adjunct distinction procedure, we refined the distinction criteria obtained during the first phase by manually inspecting the most frequent rule-types accounting for 85% of rule token expansions per non-terminal in the Penn Treebank. For an explanation on how the 85% part of rule tokens were selected, cf Section 5.2.2. Appendix C contains an overview of the ARG-ADJ distinction heuristics used in this dissertation.

Satellites that have not received an explicit argument/adjunct label based on the CCG heuristics or after the above-mentioned refinement phase, are assigned a label by default. The best experimental results were obtained by labeling all remaining satellites as adjuncts.

5.2.4 Substitution Variables: Static vs Dynamic

After flattening the original input tree into a TransBooster tree, argument and adjunct skeletons are obtained by replacing the relevant satellites by SVs, as explained in Section 4.2.4 on page 44. The translation of these simplified syntactic structures makes it possible to extract the translation of the pivot and locate the position of the satellites in target. As described in Section 4.3 on page 54, SVs can be static (SSVs) or dynamic (DSVs). In this section we will focus on the implementation of both SV types.

The experiments reported in Section 4.3.4 show that syntactic and/or lexico-semantic differences between SSVs and the constituent they replace can lead to an erroneous trans-

lation of the pivot or a wrong placement of the satellites in target. Therefore, as a first choice, we substitute the satellites with DSVs, the translations of which are obtained through late MT access. DSVs have the advantage that they share a maximal syntactic and lexico-semantic similarity with the satellites, but their retrieval in target is non-trivial, as their translation is not known in advance. This substitution leads to a *dynamic* argument skeleton and a number of *dynamic* adjunct skeletons. As a fall-back, the satellites are also substituted with appropriate SSVs, leading to a *static* argument skeleton and a number of *static* adjunct skeletons.⁸ SSVs have the advantage that they are relatively easy to track in target but their syntactic/lexico-semantic divergence with the original satellites might trigger translation errors, as shown in Section 4.3.4.

The retrieval of the translation of the pivot and the location of the satellites in target works as follows. In a first attempt, the DSV translations are matched against the translation of the *dynamic* skeletons. If a match is found for each of the DSV translations, the pivot is extracted and the position of the satellite translations is stored in memory. If there is a mismatch between one of the DSV translations and the translated skeleton, a second attempt is made by matching each of the previously established SSV translations against the translation of the *static* skeleton. Only in case this second attempt is unsuccessful, the entire pivot extraction and satellite location procedure fails (cf. 5.2.7 for more details on what happens in case no pivot can be extracted).

Instead of relying solely on DSVs or SSVs, using this back-off mechanism permits us to combine the strength of the DSV's accuracy with the SSV's retrievability. In the following section we will explain in more detail how DSVs and SSVs are generated.

5.2.4.1 Identifying Optimal Substitution Variables

The DSV of a constituent consists of the constituent's head and its simplified arguments. In other words, a constituent's DSV is the string remaining after the recursive removal of all adjuncts in the tree representation of the constituent. Removing the adjuncts leads in many cases to a considerable simplification of the original. At the same time, the presence of the head and its simplified arguments ensures a sufficient lexico-semantic resemblance

⁸Cf. Section 4.2.4 on page 44 for a schematic representation of argument/adjunct skeletons

to the original, which should avoid wrong pivot translations and erroneous satellite placements in target. Table 5.2 contains a number of example chunks and their extracted DSVs.

Satellite	→	DSV
'the stock selling pressure'	→	'the pressure'
'Wall Street professionals, including computer-guided program traders'	→	'professionals'
'the trading halt in the S&P 500 pit in Chicago'	→	'the halt'
'its remaining seven aircraft'	→	'its aircraft'
'that once did business as Merrill Lynch Commercial Real Estate'	→	'that did business'
'the potential to be so'	→	'the potential'
'the weekend preceding Black Monday in 1987'	→	'the weekend'

Table 5.2: Some examples of satellite chunks and their DSVs.

The SSV of a constituent is a simple string that, at best, shares certain syntactic characteristics with the substituted constituent. The outcome of the experiment in Section 4.3.4 showed that, even in a simplified environment, the syntactic and lexico-semantic differences between a range of SSVs and the original constituents can lead to distortions in the translation of the pivot and the placement of the satellites in target. Therefore, it is important to choose an SSV that is as similar as possible to the original. Since trying to emulate a semantic similarity between the SSV and the substituted constituent would involve an effort that goes beyond the scope of this work, we try to maximise the syntactic similarities between both.

In order to find out in which syntactic environment substitutable non-terminals need a specific SSV, we analysed the 554 most frequent rule types mentioned in Section 5.2.2.2 which cover 85% of the rule-tokens per non-terminal in sections 01–22 of the Penn Treebank. The result of this analysis is summarised in Appendix D. Each substitutable non-terminal is provided with a default SSV and, if necessary, several additional SSVs depending on the syntactic environment of the non-terminal. The appendix illustrates the SSV substitution with an example sentence for each category-environment-SSV sequence.

Note that for each of the 68 different SSV types in Appendix D, three syntactically similar but lexically different strings are available.⁹ The reason to provide SSV alternatives for

⁹These are not included in the appendix (e.g. for SSV 'the boy', the alternatives are 'the king', 'the teacher' and 'the student')

each specific category-environment sequence is to ensure correct SSV retrieval in target in the case of multiple replacement instances of the same type of satellite. If, after the pivot and satellites have been identified, multiple satellites of the same type and in an identical syntactic environment are substituted by the same SSV in a skeleton, it is not possible to locate the position of the satellites in target, given that it is highly probable that the baseline MT system will produce the same translation for identical SSV strings in the skeleton. In other words, if in (24) on page 44, $SV_{SAT_i}' = SV_{SAT_j}'$ ($1 \leq i, j \leq l+r$), a correct retrieval of the placement of SAT_i' and SAT_j' is not guaranteed.

5.2.5 Context: Static vs Dynamic

In Section 4.2.5 on page 47, we emphasised the risks of translating individual satellites out of context. Therefore, prior to being sent to the baseline MT system for translation, a satellite is embedded in a context template. This template can be *static* or *dynamic*. As is the case for Substitution Variables, the translation of dynamic context templates is determined at run-time by late MT access, while static context templates are previously translated by early MT access.

The exact nature of the dynamic context of a satellite depends on whether the satellite is an argument or an adjunct. The context for an argument satellite ARG_X is constructed based on the dynamic argument skeleton of its mother node. Given that adjuncts are not required for the correct translation of their governing node or its arguments, they can be safely omitted. Since we are interested in the translation of ARG_X , we do not substitute it in the dynamic skeleton of its mother (53), resulting in (54). In order to retrieve the translation of ARG_X , a second string is constructed, consisting of the same dynamic skeleton as before, but with ARG_X substituted with its SSV, as shown in (55).

$$(53) \quad [DSV_{ARG_1}] \dots [DSV_{ARG_i}] \textit{pivot} [DSV_{ARG_{i+1}}] \dots [DSV_{ARG_{i+r}}]$$

where DSV_{ARG_i} is the dynamic Substitution Variable for ARG_i ($1 \leq i \leq l+r$).

$$(54) \quad [DSV_{ARG_1}] \dots ARG_X \dots [DSV_{ARG_i}] \textit{pivot} [DSV_{ARG_{i+1}}] \dots [DSV_{ARG_{i+r}}]$$

$$(55) \quad [DSV_{ARG_1}] \dots SSV_{ARG_X} \dots [DSV_{ARG_i}] \textit{pivot} [DSV_{ARG_{i+1}}] \dots [DSV_{ARG_{i+r}}]$$

(53) and (54) can be represented in a simplified manner as (56) and (57), respectively.

(56) ARG_X [dynamic context] $\rightarrow ARG'_X$ [dynamic context']

(57) SSV_{ARG_X} [dynamic context] $\rightarrow SSV'_{ARG_X}$ [dynamic context']

After sending (57) to the baseline MT system, we subtract the previously known translation SSV'_{ARG_X} from the resulting string. This results in the translation of the dynamic context [dynamic context']. By subtracting [dynamic context'] from the translation of the string in (56), we obtain ARG'_X , the translation of ARG_X .

As an example, consider the sentence in Figure 4.2 on page 41, repeated below for reasons of clarity

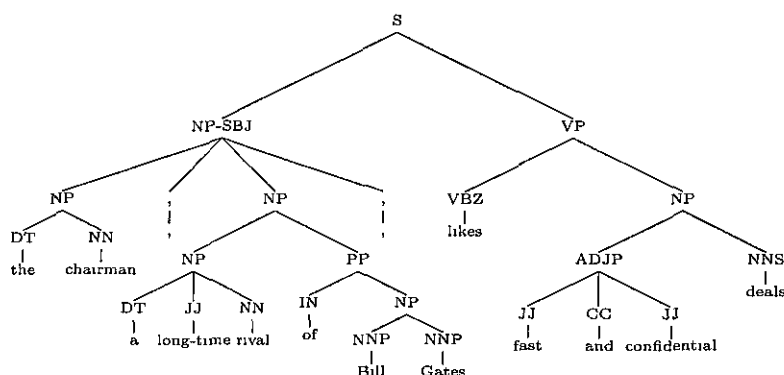


Figure 5.5: Parse tree representation of ‘The chairman, a long-time rival of Bill Gates, likes fast and confidential deals’

In order to retrieve the translation of the argument ‘fast and confidential deals’, it is embedded in the dynamic argument skeleton of its mother node, which leads to (58):

(58) [The chairman likes]_{context} fast and confidential deals. \rightarrow El presidente tiene gusto de repartos rápidos y confidenciales

We retrieve the translation of the dynamic context template ‘The man likes’ by translating the same dynamic argument skeleton, but this time containing the argument’s SSV, as shown in (59):

(59) [The chairman likes]_{context} [cars]_{SSV}. \rightarrow ‘El presidente tiene gusto de coches.’

The translation of the SSV ‘cars’ (= ‘coches’) has been previously determined by early MT access. After subtracting this string from the translation of (59), we find the translation of ‘The chairman likes’, namely ‘El presidente tiene gusto de’. By subtracting

‘El presidente tiene gusto de’ from the translation of (58), we obtain the translation of the argument ‘fast and confidential deals’, namely ‘repartos rápidos y confidenciales’.

The construction of the dynamic context of an adjunct satellite ADJ_X and the retrieval of its translation works slightly different. First, we insert ADJ_X in the dynamic skeleton of its mother node (53), which leads to the string in (60):

$$(60) \quad [DSV_{ARG_1}] \dots ADJ_X \dots [DSV_{ARG_i}] \textit{pivot} [DSV_{ARG_{i+1}}] \dots [DSV_{ARG_{i+r}}]$$

The translation of ADJ_X is obtained by retrieving the difference between the translations of (53) and (60).

As an example, consider the sentence in (61):

$$(61) \quad \text{‘Our long suit is our proven ability to operate power plants, he said.’}$$

(S (S-TPC-1 (NP-SBJ (PRP\$ Our) (JJ long) (NN suit)) (VP (VBZ is) (NP-PRD (PRP\$ our) (JJ proven) (NN ability) (S (NP-SBJ (-NONE- *)) (VP (TO to) (VP (VB operate) (NP (NN power) (NNS plants)))))))))) (, .) (NP-SBJ (PRP he)) (VP (VBD said) (SBAR (-NONE- 0) (S (-NONE- *T*-1)))))) ()

After recursively traversing the tree starting from the root node, the TransBooster algorithm arrives at the node $S\text{-}TPC\text{-}1$, which is graphically represented in Figure 5.6.

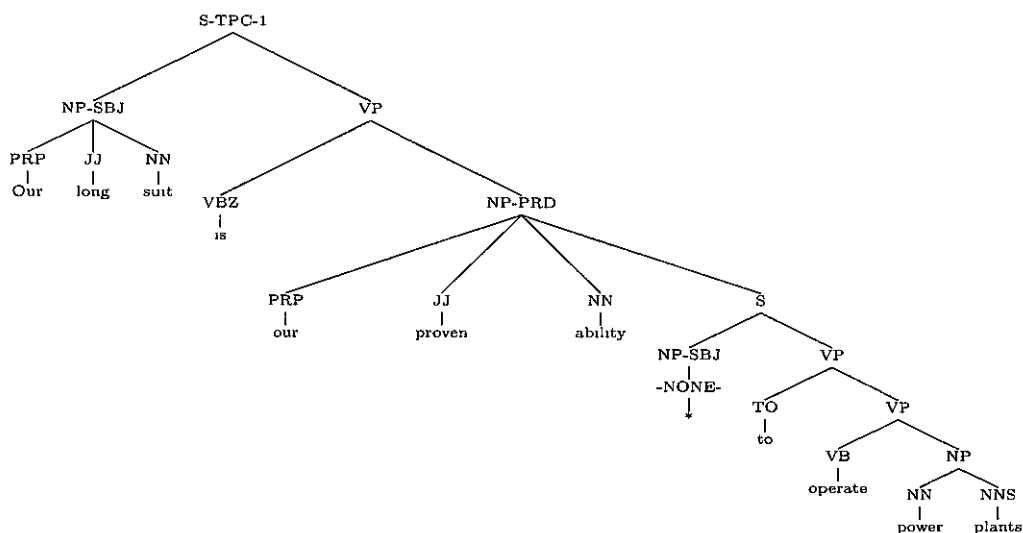


Figure 5.6: Parse tree representation of node $S\text{-}TPC\text{-}1$ in (61).

In order to retrieve the translation of the adjunct ‘to operate power plants’, it is embedded in the dynamic argument skeleton of its mother node ‘NP-PRD’, dominating

the lexical items ‘our proven ability to operate power plants’.¹⁰ Since this node does not contain any arguments, its argument skeleton consists of the pivot in isolation, represented in (62):

(62) ‘our proven ability’ → ‘nuestra habilidad demostrada.’

After inserting the adjunct into the skeleton, we obtain (63):

(63) [our proven ability]_{context} to operate power plants
→ [‘nuestra habilidad demostrada’]_{context} de operar centrales hidroeléctricas’

By retrieving the difference between the the translations of (62) and (63), we obtain the translation of the adjunct, namely ‘de operar centrales hidroeléctricas’.

Static context templates were determined by analysing the 554 most frequent rule types mentioned in Section 5.2.2.2, covering 85% of the rule-tokens per non-terminal in sections 01–22 of the Penn Treebank. The result of this analysis is summarised in Appendix E. Each non-terminal is provided with a default context and, if necessary, several additional static context templates depending on the syntactic environment of the non-terminal. The appendix illustrates the static context insertion with an example sentence for each category-environment-context sequence.

The default treatment is to embed chunks first in a dynamic context and try to extract the translation of the chunk, as described above. In case this fails, an attempt is made to extract the the chunk’s translation from a static context. If both dynamic and static context extraction fail, the chunk is translated in isolation. Note that the default backoff from *Dynamic Context*→*Static Context*→*Zero Context* can be modified depending on the specific characteristics of each chunk. For example, subject NPs need not be inserted in a context template for correct translation retrieval from English→Spanish.

¹⁰Note that the decomposition algorithm does not rely on trace information in the gold-standard Penn-II trees since this sort of detailed linguistic information is not available in the output of most statistical parsers. Penn-II functional information tags (e.g. -SBJ, -TPC, etc.) are used in the argument-adjunct distinction heuristics (cf. Appendix C), SSV selection rules (cf. Appendix D) and in the construction of Static Context Templates (cf. Appendix E).

5.2.6 Chunks and their Translation

In this section, we will discuss the back-end of the TransBooster engine, which is comprised of two modules that interact with the baseline MT system. The module *Translation* sends all strings generated by TransBooster to the MT system and retrieves their translations. The module *Chunk* ensures that all chunks to be translated are embedded in an adequate context, if necessary, and passes the generated strings on to *Translation*. After the MT system is accessed and the module *Translation* has retrieved the translations of the strings, *Chunk* extracts the chunk translations and passes them on to other modules in the TransBooster engine, which recombine the final output. This interaction is schematically represented in Figure 5.7

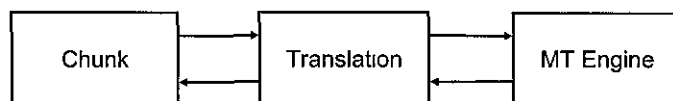


Figure 5.7: The back-end of the TransBooster Engine.

Chunk is a module containing data structures for all possible context-enhanced source-target pairs for the chunks to be sent to *Translation*. *Translation* is a module that interacts directly with the baseline MT engine by sending strings and retrieving their translations. Since all chunks are retrieved at run-time, this translation retrieval refers to late MT access. This contrasts with early MT access, in which chunks — in practice only SSVs — are translated prior to processing input by TransBooster (cf. Section 4.3.2 on page 54).

In Section 4.2 on page 39, the notion of ‘chunks’ was introduced as being the different parts that the input sentence was being decomposed into. In this section, we interpret ‘chunk’ in a broader sense: the term comprises every single item that needs to be translated in order for the algorithm to operate successfully. These items are included in Table 5.3

We will now discuss how the different types of chunks in Table 5.3 are stored and explain the default context retrieval procedure for each individual type:

Type of chunk	Default Context Retrieval
Satellites	Dynamic→Static→none
Pivots	none
Substitution Variables	none
Argument Skeletons	Dynamic→Static→none
Adjunct Skeletons	Dynamic→Static→none

Table 5.3: Chunks in module *Chunk* and their default context retrieval

Satellites Satellites are the typical chunks as introduced in Section 4.2. In most cases, it is essential to embed them in some sort of context to ensure correct translation (cf. Section 5.2.5). The data structure *Chunk* stores the satellites, retrieves their static context and constructs all necessary material for dynamic context extraction. It sends the satellite chunks to *Translation* in three different contexts: (i) in a null context (isolation), (ii) in a static context, and (iii) in a dynamic context. After retrieving the necessary translations, it attempts to extract the satellite translation from the Dynamic and Static Context translations, respectively, as explained in Section 5.2.5. In case both extractions fail, *Chunk* selects the translation of the satellite in isolation.

Pivots The translation of a pivot is obtained by extracting SV translations from the translation of the Argument Skeleton. The SVs provide the necessary context for the pivot. However, in case no pivot can be extracted from the translation of the Argument Skeleton, we want to maintain the option of retrieving the translation of the pivot in isolation. This is the reason why pivots are also sent as individual strings to the *Translation* module. In practice, retrieving the translation of pivots in isolation in case of an unsuccessful pivot extraction attempt does not lead to improvements, as might be expected.¹¹ Therefore a failed pivot extraction attempt will lead to aborting the entire decomposition process.

Substitution Variables Late MT access for Substitution Variables. both SSVs and DSVs are sent to the module *Translation* in isolation. As commented in Section 4.3.2, late MT access is the only suitable manner to retrieve the translation of a DSV. It might seem strange, though, that we are also translating SSVs at run-

¹¹Experimental results related to the program parameter `p_PivotCheck` are provided in Chapter 6

time. We do this as a safety measure: although a list of possible translations for SSVs has been determined beforehand and stored in the data structure *Substitution* (which is included in the class diagram of the TransBooster application in Appendix F), the additional SSV translation obtained at run-time will be added to this list if it is not already present. This technique also provides TransBooster with some ‘self-calibration’ to possible changes in the embedded baseline MT system.

Argument Skeletons Like proper satellites, argument skeletons are chunks that need to be embedded in a sufficient context. Therefore they receive the same treatment as satellites.

Adjunct Skeletons Like proper satellites, adjunct skeletons are chunks that need be embedded in a sufficient context. Therefore they receive the same treatment as satellites.

There are several reasons why certain chunks are translated in a zero context:

1. There is no need for additional context, e.g. in the case of a simple subject NP for English→Spanish.
2. The translation of a chunk in a zero context is the last level in the default backoff procedure (Dynamic Context → Static Context→Zero Context).
3. The chunk is used to retrieve the translation of another chunk by string subtraction. For example, the translation of a DSV is extracted from the translation of a dynamic pivot skeleton to retrieve the translation of the pivot. If the DSV were to be embedded in a context, we would somehow have to know the translation of this context as well. The only way this can be achieved is (i) by using a predefined static context (early MT access), or (ii) by translating this context at run-time (late MT access), which implies that we are simply transferring the problem to a different level, as is shown in Figure 5.8. In other words, in the case of dynamic substitutions with late MT access, at some point it is necessary to rely on the translation of an item out of context

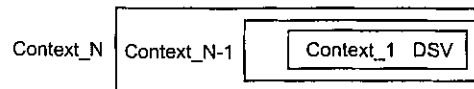


Figure 5.8: The (in theory) never-ending cycle of dynamic context template translations.

One of the essential points in the algorithm is how to determine whether a satellite chunk is ready for translation. Decomposing the input string into very small chunks has the advantage of maximal syntactic simplification, but overall translation might not improve due to context issues. On the other hand, a limited decomposition in larger chunks will not suffer that much from context deterioration but will lead to less syntactic simplification. Due to the average time needed for an experiment-evaluation cycle,¹² it is not possible to determine a different cut-off threshold for each different category in each different syntactic setting. In the current implementation, we maintain the same cut-off point N for all types of satellite chunks. This cut-off point depends on the number of lexical items that the node representation of the chunk dominates. If the node dominates fewer or the same number of lexical items than the threshold N , it is translated in its entirety, embedded in a context template if necessary. If the node dominates more than N lexical items, it is subjected to decomposition. The threshold N is one of the program's parameters: its optimal value depends on the baseline MT system used and was established empirically, for each different baseline MT system, by tuning the program parameter `p_ChunkLength`, as will be further explained during the discussion of experimental results in Chapter 6.

In the algorithm presented in Section 5.2.8, the baseline MT system is accessed at several different stages during the decomposition of each individual sentence. This is a simplified representation of what really happens. Sending a string to the MT system, executing the translation and retrieving the translated output consumes a certain amount of time, depending on the length of the string, the system used and the interface to TransBooster. Given that the decomposition of one single sentence can easily lead to hundreds of different strings (satellites, pivots, SVs and skeletons) to be translated, in practice, continuous MT access would be too time-consuming. Therefore, the TransBooster algorithm is split into three different parts, as is graphically represented in Figure 5.9:

¹²Depending on the MT system used, between 25–35 minutes per experiment-evaluation cycle

1. Decomposition: all sentences in the input file are decomposed. All resulting different individual chunks are written to a data file.
2. Translation: the data file is translated by the baseline MT system.
3. Recomposition: the translations of all chunks are retrieved and the output sentences are composed

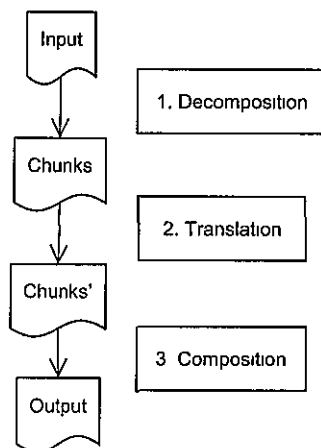


Figure 5.9: The three stages in a TransBooster run.

This way, the MT system is only accessed once per TransBooster run. The module *Translation* contains data structures that ensure that no duplicate strings are translated. The module also performs a number of necessary pre-processing steps on the strings that are being sent to the MT engine. For example, each candidate string for translation must commence with a capital letter and end with a dot. Failure to do so might result in a distorted translation, as is shown in the examples (64) and (65):

- (64) "The man is sleeping," says Mr. Zurkuhlen. → "El hombre está durmiendo", el Sr. Zurkuhlen dice
 "The man is sleeping," says Mr. Zurkuhlen → *"El hombre está durmiendo", decir al Sr. Zurkuhlen
- (65) 'I'm not going to worry about the dog.' → 'No voy a preocuparme por el perro.'
 'i'm not going to worry about the dog.' → *'I no va para preocuparse por el perro.'

Seemingly trivial details like the ones in (64) and (65) can lead to important changes in

translation quality. In (64), the translation of the second sentence contains an uninflected form of the main verb ('decur') in the wrong place. In (65), the output is incomprehensible due to a mimicked subject ('i'), a wrong inflection of the main verb ('va') and an erroneous preposition ('para'). The module also performs certain operations regarding punctuation and whitespace that might have been distorted during the building of a skeleton or after inserting a chunk into its context.

5.2.7 Safety Measures

During the decomposition of a chunk, a number of problems can arise that cause the decomposition process to abort. If such problems occur, it is always possible, as a back-off measure, to translate the chunk in its entirety. The main two problems that trigger this back-off measure are the following:

1. The translation of an SV is not found in the translated skeleton. This occurs if both the retrieval of DSVs and SSVs in the translated skeletons is unsuccessful. In this case, it is impossible to extract the translation of the pivot.
2. If the pivot is retrieved via SSV substitution, we verify the presence of the extracted pivot translation in the translation of the dynamic argument skeleton. Since the dynamic argument skeleton shares more syntactic/lexico-semantic similarities with the original, a mismatch might indicate an erroneous translation of the pivot in the static argument skeleton. In this case, we deem the extracted pivot translation unreliable.

If it was impossible to extract a translation of the pivot or if the extracted pivot is considered unreliable, there are two back-off alternatives:

1. Abort the decomposition process and translate the entire node as an indivisible unit.
2. Translate the pivot in isolation and continue the decomposition process.

Although both choices exist as a program parameter¹³, experiments (reported in Chapter 6) show that the first back-off alternative yields much better results, which is to be expected.

¹³Parameter `p_PivotCheck`, as will be further explained in Chapter 6

5.2.8 Algorithm

Figure 5.10 shows the standard TransBooster algorithm (TB_{MarkI}) in pseudo-code. The operation of the algorithm is illustrated with a simple example.

```
1  Input = parsed sentence,
2  S = Tree data structure of Input;
3  Recursive head/arg/adj annotation of nodes S;
4  QUEUE = {S};
5  While (QUEUE not empty) {
6    Node N = shift QUEUE;
7    If (N OK for translation) {
8      translate N (in context);
9    }
10   else {
11     flatten N into TransBooster tree;
12     - find pivot N;
13     - find satellites N;
14     find SVs for all satellites;
15     build skeletons;
16     translate SVs;
17     translate skeletons;
18     find translation pivot;
19     if (translation pivot not OK) {
20       translate N (in context);
21       break;
22     }
23     track location satellites in target;
24     add all satellites to QUEUE;
25   }
26 }
27 Recompose(S) where
28 Recompose(N) {
29   for (all satellites of N) {
30     sort all satellite SVs and pivot with respect to
31     their position in target;
32     if (satellite OK for translation) {
33       replace SV satellite with translation satellite;
34     }
35     else {
36       recompose satellite;
37     }
38   }
39 }
```

Figure 5.10: The standard TransBooster algorithm (TB_{MarkI}) in pseudo-code

5.2.8.1 Worked Example

In this section, we illustrate the standard TransBooster algorithm (TB_{MarkI}) on the Penn-II sentence ‘One week later, Leonard H. Roberts, president and chief executive officer of

Arby's, was fired in a dispute with Mr. Posner'. The baseline MT system is LogoMedia, the language pair English→Spanish. The output of the example sentence by the baseline system is shown in (66):

- (66) 'Uno semana después, Leonard H Roberts, presidente y funcionario en jefe ejecutivo de Arby's, fue disparado en una disputa con el Sr Posner.'

The main problem in this translation is that LogoMedia's transfer module has erroneously selected 'fired' → 'disparado' (= 'shot') instead of the correct 'fired' → 'despedido' (= 'sacked').

The input to the decomposition algorithm is (67)

- (67) (TOP (S (ADVP-TMP (NP (CD One) (NN week)) (RB-later)) (, ,) (NP-SBJ-1 (NP (NNP Leonard) (NNP H) (NNP Roberts)) (, ,) (NP (NP (NP (NN president)) (CC and) (NP (JJ chief) (JJ executive) (NN officer))) (PP (IN of) (NP (NNP "Arby") (POS-'s)))) (, ,) (VP (VBD 'was) (VP (VBN fired) (NP (-NONE- *-1)) (PP-LOC (IN in) (NP (NP (DT a) (NN dispute)) (PP (IN with) (NP (NNP Mr.) (NNP Posner))))))))) (,))

Step 1

The algorithm finds the pivot 'was fired' and the satellites [One week later]_{ADJ1}, [Leonard H Roberts, president and chief executive officer of Arby's]_{ARG1} and [in a dispute with Mr. Posner]_{ADJ2}. This leads to the flattened structure in Figure 5.11.

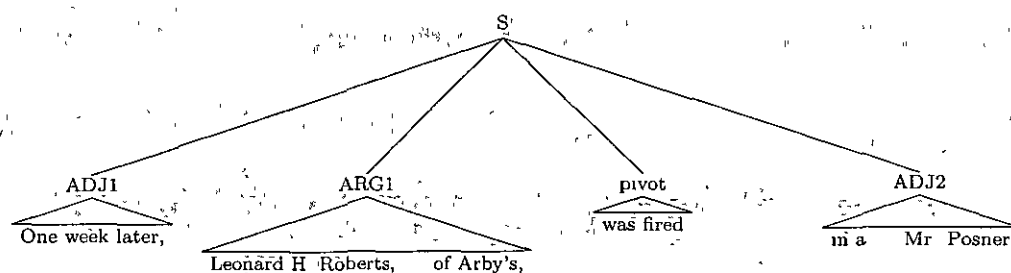


Figure 5.11 TransBooster tree representation of (67).

TransBooster replaces the argument satellite by the DSV 'Leonard H. Roberts' and sends the argument skeleton in (68) to the baseline MT engine. Since we have determined the translation of the DSV 'Leonard H Roberts' at runtime, it is possible to extract the translation of the pivot ('fue despedido') and locate the position of the argument satellite

in target

(68) '[Leonard H. Roberts] [was fired].' → '[Leonard H Roberts] [fue despedido]'

Note that the simplified syntactic structure of the argument skeleton in (68) already leads the baseline MT system to correctly translate 'fired' as 'despedido'.

Next, two adjunct skeletons are constructed, one for ADJ1 'One week later,' and one for ADJ2 'in a dispute with Mr. Posner', by inserting the DSVs for both adjuncts, one by one, in the argument skeleton.

(69) '[One week later,]_{ADJ} [Leonard H. Roberts] [was fired]' → '[Uno semana después,]_{ADJ}
[Leonard H Roberts] [fue despedido]'

'[Leonard H Roberts] [was fired] [in a dispute]_{ADJ}.' → '[Leonard H Roberts] [fue despedido] [en una disputa]_{ADJ}.'

From the translation of both adjunct skeletons in (69), we deduce the position of the adjuncts in target. After this first step, we have found the translation of the pivot and have determined the location of all satellites in target.

Step 2

In a second step, the algorithm investigates the first satellite ('One week later'), and decides that it is simple enough for translation, since it contains fewer than the optimal threshold N lexical items¹⁴ Before sending the satellite to the baseline MT system for translation, it is embedded in a dynamic context as explained in Section 5.2.5. This leads to the string in (70):

(70) '[One week later,] [Leonard H Roberts]_{DSV_{ARG1}} [was fired]_{pivot}.' → '[Uno semana después,] [Leonard H Roberts]_{DSV_{ARG1}} [fue despedido]_{pivot}.'

Since we have already found the translation of the pivot ('fue despedido') and since the translation of the DSV 'Leonard H. Roberts' was determined by late MT access, it is possible to deduce 'Uno semana después' as the translation of the satellite 'One week later' from (70).

¹⁴ N was determined empirically for each baseline MT system by tuning parameter `p_ChunkLength`. In the case of LogoMedia, optimal results were obtained with $N = 5$.

Step 3

Assume, for the sake of simplicity, that the second satellite, ('Leonard H. Roberts, president and chief executive officer of Arby's') is considered ready for translation. Like the first satellite, it is embedded in a dynamic context. Since 'Leonard H. Roberts, president and chief executive officer of Arby's' is the only argument, its dynamic context consists exclusively of the pivot, as is shown in (71):

- (71) 'Leonard H. Roberts, president and chief executive officer of Arby's' [was fired]_{pivot}.
→ Leonard H Roberts, presidente y funcionario en jefe ejecutivo de Arby's, [fue disparado]_{pivot}.

Note that in this string, the pivot once again obtains the erroneous translation 'fue disparado'. Since the previously established pivot translation 'fue despedido' cannot be found in the translation of (71), the retrieval of the translation of the second satellite fails. Therefore, we back off to the construction of the static context, as is shown in (72):

- (72) 'Leonard H. Roberts, president and chief executive officer of Arby's' [is sleeping]_{context}.
→ Leonard H Roberts, presidente y funcionario en jefe ejecutivo de Arby's, [está durmiendo]_{context}.

This time, the string 'está durmiendo', previously established by early MT access, is found in the translation of (72). By string subtraction, we obtain the translation of the second satellite 'Leonard H Roberts, presidente y funcionario en jefe ejecutivo de Arby's'.

Step 4

The last satellite, 'in a dispute with Mr. Posner', contains 6 lexical items. Since this number > the optimal threshold ($N=5$) established for LogoMedia, the satellite is subject to further decomposition.¹⁵ Let's assume, in order to keep this example decomposition clear and simple, that the satellite is not further decomposed and is considered ready for translation. It is then embedded in a dynamic context template and sent to the baseline MT system for translation, as is shown in (73):

¹⁵Pivot = 'in'. ARG = 'a dispute with Mr. Posner'

- (73) '[Leonard H Roberts]_{DSV_{ARG1}} [was fired]_{pivot} [in a dispute with Mr. Posner.]' →
 '[Leonard H Roberts]_{DSV_{ARG1}} [fue despedido]_{pivot} [en una disputa con el Sr. Posner.]'

Since we have already found the translation of the pivot ('fue despedido') and since the translation of the DSV 'Leonard H Roberts' was determined by late MT access, it is possible to deduce 'en una disputa con el Sr. Posner' as the translation of the satellite 'in a dispute with Mr. Posner' from (73)

Step 5

After all satellites have been decomposed and translated, the algorithm, in a final step, composes the output by stitching together the obtained translations in the target locations found by the SV translations. After step 1, we found the relative ordering of satellites around the pivot as shown in (74):

- (74) [SV_{ADJ1}] [SV_{ARG1}] [pivot] [SV_{ADJ2}]

By placing the translations of the satellites in their correct slot, we obtain the final result in (75):

- (75) 'Uno semana después, Leonard H Roberts, presidente y funcionario en jefe ejecutivo de Arby's, fue despedido en una disputa con el Sr. Posner.'

The result in (75) improves on the original translation of the baseline MT system in (66), since the reduction of syntactic complexity forced the baseline MT system to correctly translate 'fired' as 'despedido' instead of the erroneous 'disparado'.

5.3 TransBooster Mark II

A precondition for the algorithm in Section 5.2.8 to function correctly is that the translation of the pivot is not split in target. In the argument skeleton of TB_{MarkI} in (76), the translation of the pivot *pivot* is treated as an indivisible unit with respect to which the placement of the satellites in target is calculated:

$$(76) \quad [SV_{ARG_1}] \cdot [SV_{ARG_l}] \textit{pivot} [SV_{ARG_{l+1}}] \dots [SV_{ARG_{l+r}}] \rightarrow$$

$$[SV'_{ARG_1}] \quad [SV'_{ARG_l}] \textit{pivot}' [SV'_{ARG_{l+1}}] \dots [SV'_{ARG_{l+r}}]$$

where SV_{ARG_i} is the simpler string substituting ARG_i , ($1 \leq i \leq l+r$).

This approach would lead to problems in sentences in which the translation of the pivot is split in two or more parts, as is illustrated in (77), translated from English into German by LogoMedia:

$$(77) \quad [\textit{The man}]_{SV_{ARG_1}} [\textit{has eaten}]_{\textit{pivot}} [\textit{an apple}]_{SV_{ARG_2}} \cdot \rightarrow [\textit{Der Mann}]'_{SV_{ARG_1}} [\textit{hat}]'_{\textit{pivot}_1}$$

$$[\textit{einen Apfel}]'_{SV_{ARG_2}} [\textit{gegessen}]'_{\textit{pivot}_2}.$$

In the construction in (77), typical of most Germanic languages, the pivot $[\textit{has eaten}]_{\textit{pivot}}$ is split in two parts in target ($[\textit{hat}]'_{\textit{pivot}_1}$ and $[\textit{gegessen}]'_{\textit{pivot}_2}$), which makes it impossible to determine the location of the translation of the satellites according to the algorithm in Section 5.2.8. In order to be able to handle cases with a split pivot translation, we implemented an alternative, simplified version of the $TB_{\textit{MarkI}}$ algorithm, relying solely on string replacements of satellite SVs in target.

5.3.1 Mark I vs. Mark II

The flattening of the input tree into a TransBooster tree with one pivot and several satellite nodes proceeds in the same manner as explained in Section 4.2.1 on page 40, resulting in the construction represented in Figure 5.12:

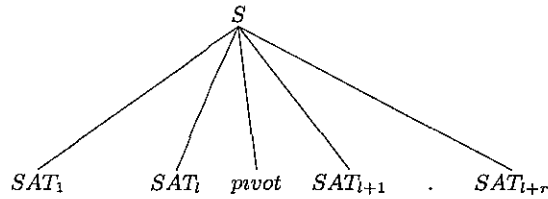


Figure 5.12: Input Chunk S into decomposition algorithm of $TB_{\textit{MarkII}}$

Instead of working with two substitution skeletons, one for arguments and one for adjuncts (cf Section 4.2.4 on page 44), only *one* skeleton is constructed, in which a number of satellites are substituted. The exact nature of the satellites to be substituted is determined before a TransBooster run by setting parameters regarding its syntactic category and the number of its leaf nodes. The other satellites remain unchanged in

the skeleton. For example, in (78), SAT_1 and SAT_{l+r} are substituted by their SV. The remainder of the skeleton consists of the pivot and the original coverage of the other satellites.

(78) $[SV_{SAT_1}] \dots [SAT_l] \textit{pivot} [SAT_{l+1}] \dots [SV_{SAT_{l+r}}]$

The string in (78) is sent to the baseline MT engine, leading to the translation in (79):

(79) $XXX [SV'_{SAT_1}] YYY [SV'_{SAT_{l+r}}] ZZZ.$

where XXX, YYY and ZZZ are sequences of strings comprising the translation of the pivot and the satellites that have not been substituted.

As an example, consider the sentence in (80):

(80) 'Her friend David, whose parents kept reminding him he was unwanted, slept on a narrow bed wedged into her parents' bedroom, as though he were a temporary visitor.'

In a scenario in which we want to substitute only NP and PP satellites with a lexical coverage greater than 4 words by an SSV, the flattened TransBooster tree in (81) would lead to the skeleton in (82):

(81) '[Her friend David, whose parents kept reminding him he was unwanted,]_{ARG1} [slept]_{pivot} [on a narrow bed wedged into her parents' bedroom,]_{ADJ1} [as though he were a temporary visitor]_{ADJ2}'

(82) '[The boy]_{SSV_{ARG1}} slept [in the house]_{SSV_{ADJ1}} as though he were a temporary visitor.

The string in (82) is a real-world example of (78). The translation of this string by the baseline MT system is (83), which is an example of (79).

(83) '[El niño]_{SSV'_{ARG1}} durmió [en la casa]_{SSV'_{ADJ1}} como si era una visita temporal.'

If the substituted satellites SAT_1 and SAT_{l+r} are deemed simple enough for translation, they are embedded in a simplified context as described in Section 4.2.5 and sent to the baseline MT system for translation. If the substituted satellites SAT_1 and SAT_{l+r} are deemed too complex for translation, the entire procedure is recursively applied to the satellites, i.e. the satellite chunks themselves are decomposed into a pivot and satellites, which in turn are examined for translatability.

Let us suppose, for the sake of simplicity, that [Her friend David, whose parents kept reminding him he was unwanted,]_{ARG1} and [on a narrow bed wedged into her parents' bedroom,]_{ADJ1} in (81) are considered ready for translation. By embedding both satellites in a static context and sending the resulting strings to the baseline MT system, we obtain the translations in (84).

$$(84) \quad \begin{aligned} & [\text{Her friend David, whose parents kept reminding him he was unwanted,}]_{ARG1} [\text{is sleeping.}]_{context} \rightarrow [\text{Su amigo David, cuyos padres guardaron recordarlo que era no deseado,}]'_{ARG1} [\text{est durmiendo.}]'_{context} \\ & \text{The man is sleeping}]_{context} [\text{on a narrow bed wedged into her parents' bedroom,}]_{ADJ1} \\ & \rightarrow [\text{El hombre est durmiendo}]'_{context} [\text{en una cama angosta calzada en el dormitorio de sus padres.}]'_{ADJ1} \end{aligned}$$

Since we have established the translation of the SVs [SV'_{SAT_1}] and [$SV'_{SAT_{i+r}}$], either by early or by late MT access, we obtain the final result by replacing the translations of the SVs by the translations of the corresponding satellites in (79). In our example, we replace [El nio]_{SSV'_{ARG1}} and [en la casa]_{SSV'_{ADJ1}} in (83) by [Su amigo David, cuyos padres guardaron recordarlo que era no deseado,]'_{ARG1} and [en una cama angosta calzada en el dormitorio de sus padres.]'_{ADJ1} respectively, leading to the final result in (85):

$$(85) \quad \begin{aligned} & [\text{Su amigo David, cuyos padres guardaron recordarlo que era no deseado,}]'_{ARG1} \text{durmio} \\ & [\text{en una cama angosta calzada en el dormitorio de sus padres.}]'_{ADJ1} \text{ como si era una visita temporal.} \end{aligned}$$

Note that, contrary to the algorithm in TB_{MarkI} , we do not explicitly distinguish between arguments and adjuncts, the reason being that recomposition in TB_{MarkII} relies only on string replacement and does not compose the output by placing the translations of the satellites in their appropriate target location with respect to the translation of the pivot, as is done in TB_{MarkII} .

5.3.2 Algorithm

Figure 5.13 shows the simplified TransBooster algorithm (TB_{MarkII}) in pseudo-code. The main differences between the original TB_{MarkI} algorithm in Figure 5.10 on page 91 and the simplified TB_{MarkII} algorithm in Figure 5.13 are:

```

1  Input = parsed sentence;
2  S = Tree data structure of Input;
3  Recursive head/arg/adj annotation of nodes S;
4  QUEUE = {S};
5  While (QUEUE not empty) {
6    Node N = shift QUEUE;
7    If (N OK for translation) {
8      translate N (in context);
9    }
10   else {
11     flatten N into TransBooster tree;
12     - find pivot N;
13     - find satellites N;
14     substitute certain satellites;
15     select candidates for recursion
16     from substituted satellites;
17     add candidates to QUEUE;
18   }
19 }
20 Recompose(S) where
21 Recompose(N) {
22   for (all substituted satellites of N) {
23     if (satellite OK for translation) {
24       replace SV satellite in translation skeleton
25       with translation satellite;
26     }
27     else {
28       recompose satellite,
29     }
30   }
31 }

```

Figure 5 13: The simplified TransBooster algorithm (TB_{MarkII}) in pseudo-code.

1. Where TB_{MarkI} makes a distinction between argument and adjunct skeletons, TB_{MarkII} only uses one type of skeleton in which all satellites are replaced by their SVs.
2. During a run of TB_{MarkII} , it is possible to determine which satellites are substituted and which are recursed into in a subsequent run. In TB_{MarkI} , all satellites are substituted and all substituted satellites dominating a certain number of leaf nodes are candidates for recursion.
3. Recomposition in TB_{MarkII} is based on string replacement in the translated skeleton. Recomposition in TB_{MarkI} is performed by ‘stitching together’ the retrieved translations of all satellites around the translation of the pivot.

The advantages of TB_{MarkII} over TB_{MarkI} are: (i) TB_{MarkII} is able to deal with split pivots in target, and (ii) in TB_{MarkII} , it is possible to specify exactly which satellites are to be substituted, whereas in TB_{MarkI} , *all* satellites that contain more than a certain number of leaf nodes are substituted. Unlike TB_{MarkI} , the simplified string insertion algorithm of TB_{MarkII} does not need a full syntactic parse as input, but only requires the correct identification of the substitutable constituents. Therefore, it is possible to use partial parsing or chunking to produce the input for TB_{MarkII} , which could be an interesting alternative for input languages for which no high-quality full parsers have been developed. The disadvantage of TB_{MarkII} with respect to TB_{MarkI} is that skeletons in TB_{MarkII} necessarily have to contain both arguments and adjuncts. Therefore, TB_{MarkII} provides less room for syntactic complexity reduction than TB_{MarkI} .

5.4 Summary

This chapter contains the technical details of the TransBooster architecture. We have explained both the standard TransBooster algorithm (Section 5.2 TransBooster Mark I) and the simplified TransBooster strategy (Section 5.3 TransBooster Mark II), illustrating each concept with one or more examples.

In general, TransBooster tackles the complexity reduction problem by (i) replacing complex constituents with simple substitution variables, (ii) omitting adjuncts in argument skeletons (only for TB_{MarkI}), and (iii) sending only short, simple chunks for translation to the baseline MT systems.

In the next chapter, we will analyse the experimental results of TransBooster interfaced with three RBMT systems and two data-driven systems.

Chapter 6

Experimental Results and Analysis

6.1 Introduction

In this chapter we present and analyse the results of the TransBooster architecture interfaced with the baseline systems introduced in Chapter 3. Section 6.2 contains results on the RBMT systems LogoMedia, Systran and SDL. In Section 6.3, we analyse the results of TransBooster interfaced with two data-driven MT systems: a phrase-based SMT system and a marker-based EBMT system.

6.2 Results for Rule-based MT

6.2.1 Experimental setup

This section contains an analysis of the results of TransBooster interfaced with the three rule-based systems used in this thesis. We first explain TransBooster's program parameters and present automatic evaluation results of TransBooster Mark I with optimal parameter settings on the pre-parsed 800-sentence test set described in Chapter 3. We argue that automatic evaluation metrics alone might not be sensitive enough to accurately measure the performance of TransBooster and include a manual evaluation on 200 sentences, randomly selected from the test set, for each of the baseline systems. We explain the most important areas of improvement with a number of examples and analyse why some sentences receive a worse translation despite a correct complexity reduction. We then investigate

the impact of parser-based input on the algorithm by parsing the 800-sentence test set with (Charniak, 2000) and (Bikel, 2002). Finally, we analyse the results of the alternative, simplified TransBooster architecture (TB_{MarkII}) presented in the previous chapter.

6.2.2 Experiments with TransBooster Mark I

TransBooster has five different program parameters, which were explained in previous chapters and are summarised in Table 6.1. Table 6.2 contains the optimal parameter settings per baseline MT engine. These are the settings that were used to produce the automatic evaluation results reported in the following section.

Name	Value	Definition	Pages
p_ChunkLength	positive integer	Recursion threshold. Its value is the minimal number of lexical items that a node has to contain in order to be eligible for decomposition (cf. Section 5.2.6)	50, 88, 93
p_PivotLength	positive integer	Threshold of pivot length. Its value is the maximal number of lexical items that a pivot can contain, if constructed by Default Tree Flattening (cf. Section 5.2.2.1).	73
p_PivotAttach	positive integer	Its value is the maximal number of leaf nodes that a satellite, adjacent to the pivot, can contain in order to be included in the pivot (cf. Section 5.2.2)	72
p_PivotCheck	boolean	If true, verify the presence of the extracted pivot in the translation of the Dynamic Argument Skeleton and abort decomposition if not found. (cf. Section 5.2.7).	86, 90
p_SatDefault	string	If 'arg', the default assignment of a satellite is <i>argument</i> . Else it is <i>adjunct</i> . Argument-adjunct distinction based on CCG-induced rules takes preference over the default assignment (cf. Section 5.2.3)	78

Table 6.1: TransBooster program parameters, their definition and the pages in the thesis where they are explained

Name	LogoMedia	Systran	SDL
p_ChunkLength	5	4	5
p_PivotLength	4	4	4
p_PivotAttach	2	3	2
p_PivotCheck	true	true	true
p_SatDefault	'adj'	'adj'	'adj'

Table 6.2: Optimal parameter settings per baseline MT system.

6.2.2.1 Automatic evaluation

Table 6.3 contains the results for the optimal settings on the Penn-II Treebank 800-sentence test set. TransBooster improves between 0.7%-1.7% relative BLEU score, 0.5%-1.0% NIST score and 0.1%-0.5% GTM score, depending on the baseline MT system used.¹

	BLEU	NIST	GTM
LogoMedia	0.3140	7.3272	0.5627
TransBooster	0.3188	7.3709	0.5658
Percent of Baseline	101.5%	100.5%	100.5%
Systran	0.3003	7.1674	0.5553
TransBooster	0.3024	7.2142	0.5582
Percent of Baseline	100.7%	100.6%	100.5%
SDL	0.3039	7.2735	0.5657
TransBooster	0.3093	7.3490	0.5663
Percent of Baseline	101.7%	101.0%	100.1%

Table 6.3 TransBooster results on the 800-sentence test set with optimal parameters.

When carrying out the experiments, we realised that the reference translations for the 800-sentence test set were slightly biased towards the baseline MT systems, since the translators who produced the reference set were presented the output of one of the baseline systems, in random order, and were asked to use parts of the MT output if they considered it useful, as was explained in Section 3.4.2. Given that four different baseline MT systems were used for 1/4 of the entire test set (200 sentences), it would seem natural that the translations of each set of 200 sentences would contain a slight bias towards the baseline MT system used. To test this hypothesis, we removed the 200 possibly biased

¹The statistical significance of these results, and the other results in this chapter, was established in a 95% confidence interval by using the BLEU/NIST resampling toolkit described in (Zhang and Vogel, 2004) <http://projectile.is.cs.cmu.edu/research/public/tools/bootStrap/tutorial.htm>

sentences for LogoMedia, Systran and SDL from the original 800-sentence test set, thus producing three ‘unbiased’ 600-sentence reference test sets, one for each of the different baseline MT systems. For example, the ‘unbiased’ 600-sentence test set for LogoMedia was constructed by removing the 200 sentences that were translated by LogoMedia from the original 800-sentence test set that was presented to the translators.

	BLEU	NIST	GTM
LogoMedia	0.2830	6.8555	0.5391
TransBooster	0.2907	6.9082	0.5442
Percent of Baseline	102.7%	100.7%	100.9%
Systran	0.2708	6.7244	0.5368
TransBooster	0.2745	6.7816	0.5399
Percent of Baseline	101.4%	100.8%	100.6%
SDL	0.2823	6.8917	0.5473
TransBooster	0.2904	6.9878	0.5496
Percent of Baseline	102.8%	101.4%	100.4%

Table 6.4 TransBooster results on the three 600-sentence test sets with optimal parameters.

Table 6.4 contains the results of TransBooster on the three ‘unbiased’ 600-sentence test sets. In comparison with Table 6.3, the relative BLEU scores increase from 101.5% to 102.7% for LogoMedia, from 100.7% to 101.4% for Systran and from 101.7% to 102.8% for SDL. NIST scores increase from 100.5% to 100.7% for LogoMedia, from 100.6% to 100.8% for Systran and from 101.0% to 101.4% for SDL. GTM scores increase from 100.5% to 100.9% for LogoMedia, from 100.5% to 100.6% for Systran and from 100.1% to 100.4% for SDL.

In Section 5.2.7 on page 90, we explained the safety measure that enables TransBooster to have an input chunk translated in its entirety instead of proceeding with decomposition when there is an indication that something has gone wrong in the decomposition process. This back-off measure can be activated by setting the parameter `p_PivotCheck`. Table 6.5 shows the impact on the scores in Table 6.3 of the deactivation of `p_PivotCheck`.

The results in Table 6.5 clearly show that the back-off procedure has a positive impact on the scores. The size of this impact depends on the baseline MT system used. Backing off is more beneficial in the case of Systran and LogoMedia than it is for SDL. Since

	p_PivotCheck	BLEU	NIST	GTM
TB LogoMedia	true	0.3188	7.3709	0.5658
TB LogoMedia	false	0.3144	7.3049	0.5619
false vs. true		98.6%	99.1%	99.3
TB Systran	true	0.3024	7.2142	0.5582
TB Systran	false	0.2934	7.1303	0.5534
false vs. true		97%	98.8%	99.1%
TB SDL	true	0.3093	7.3490	0.5663
TB SDL	false	0.3089	7.3408	0.5662
false vs. true		99.9%	99.9%	99.9%

Table 6.5: Impact of parameter p_PivotCheck on the results in Table 6.3

we do not have access to the internal workings of the baseline MT systems, we can only make a calculated guess to why this is the case. It is likely that the SDL engine is less context-sensitive than the other two RBMT systems, i.e. either its lexicon contains fewer alternative translations or its analysis module produces parses with less variation than Systran and LogoMedia.

6.2.2.2 Manual Evaluation

Automatic evaluation measures are useful to compare MT systems of the same MT paradigm when a large enough evaluation corpus is available (Callison-Burch et al., 2006). As pointed out in Section 3.4.1 on page 27, automatic metrics are not, and were never designed to be, a *substitute* for human assessment of translation quality. Moreover, since all three automatic evaluation methods that we used are based on string-based similarity metrics, one can ask the question whether they are sensitive enough to adequately capture the differences between two relatively similar MT outputs.

TransBooster ultimately relies on the output translations produced by a baseline MT system. Therefore, although it is certainly possible for TransBooster to help the system improve its own translations (as has been shown in the numerous examples in this thesis), in rare cases the output of TransBooster and the baseline MT system will be radically different. In addition, the necessary back-off measures will lead TransBooster in a number of cases to suspend decomposition at the root node, in which cases TransBooster will produce exactly the same translation as the baseline MT system. Table 6.6 contains the number of times that the back-off procedure was invoked at the root node (in the optimal

settings as reported in Table 6.2) for the 800-sentence test set. In these cases (23.6% for LogoMedia, 29.4% for Systran and 20.4% for SDL), the input sentence is translated in its entirety by the baseline MT system.

	LogoMedia	Systran	SDL
Absolute Nr	189	235	163
% of 800-sentence test set	23.6%	29.4%	20.4%

Table 6.6 Proportion of sentences per MT engine (in the optimal setting) in which the back-off procedure is invoked at the root node. Invoking back-off at the root will disable decomposition for the entire sentence, so that the entire input is translated *as is* by the baseline MT system.

Table 6.7 shows the percentages of *lexical* differences between the output of TransBooster and the baseline MT system for all 800 sentences in the test corpus and for the non-backed-off sentences (cf. Table 6.6), i.e. the sentences in which the TransBooster decomposition algorithm was invoked. The figures in Table 6.7 only represent *lexical* differences and do not take word order into account: they were calculated by considering each TransBooster and baseline MT output sentence as a *bag of words*, as is shown in equation (6.1):

$$P = \frac{\# \text{ words in TB output with no exact match in baseline MT output}}{\# \text{ words in TB output}} \times 100 \quad (6.1)$$

	LogoMedia	Systran	SDL
P for non-backed-off sentences	4.84%	5.41%	4.26%
P for all sentences	3.76%	3.73%	3.42%

Table 6.7: Percentages of different words between TransBooster and the baseline systems on the 800-sentence test set. Figures are provided for the entire test set and for those sentences for which the back-off procedure was invoked. P is explained in Formula 6.1.

The figures in Table 6.7 show that invoking TransBooster, on average, will not radically change the lexical structure of the original output produced by the baseline MT system. Over the entire 800-sentence test corpus, TransBooster produces 3.76% lexical differences

compared to the output of LogoMedia, 3.73% compared to Systran and 3.42% compared to SDL. Since these differences are not very pronounced, it would be prudent to corroborate the automatic evaluation scores with a manual evaluation.

The test set for the manual evaluation was constructed by randomly selecting 200 sentences out of the pool of sentences for which TransBooster produced a result different from the original baseline MT output. Table 6.8 contains the number of sentences for which the TransBooster decomposition procedure produced a result different from the baseline MT output. For LogoMedia, Systran and SDL, this pool contains 325, 368 and 367 sentences respectively. This means that for LogoMedia, Systran and SDL, TransBooster produced the same result as the baseline MT system in 475, 432 and 433 sentences respectively, either because the backoff procedure was invoked at the root node or because the actual TransBooster decomposition did not lead the baseline MT systems to change their original translation. We chose to select sentences for manual evaluation exclusively from the pool of different sentences of Table 6.8 in order to maximise the coverage of the manual evaluation, since it is straightforward to extrapolate the manual evaluation results on 200 sentences to approximate a manual evaluation of the entire 800-sentence test set by taking into account the amount of sentences for which TransBooster produced the same result as the baseline MT systems, as we will explain below.

	LogoMedia	Systran	SDL
Nr. of different sentences	325	368	367
% of 800-sentence test set	40.6%	46.0%	45.9%

Table 6.8: Number of TransBooster output sentences that are different from the baseline MT system's output.

The resulting 600 evaluation units (3×200 different TransBooster vs. baseline MT outputs) were randomly distributed between eight native Spanish linguistic experts with previous experience in MT. The experts were asked to produce a comparative evaluation by selecting, for each evaluation unit they were presented², the better translation (if any), both in terms of accuracy and fluency. We explained the rationale for this testing procedure in Section 3.4.1.5. Tables 6.9 and 6.10 show the results of the manual evaluation.

²evaluation unit = <TransBooster output vs. Baseline MT output>

	TB vs LogoMedia			TB vs. Systran			TB vs SDL		
	B	S	W	B	S	W	B	S	W
Fluency %	36.50	38.50	25.00	27.00	48.00	25.00	40.50	36.00	23.50
Accuracy %	27.50	48.50	24.00	26.00	47.00	27.00	38.00	40.50	21.50

Table 6.9: Comparative results of the manual evaluation of TransBooster vs LogoMedia, Systran and SDL on 200 different output sentences. B = better, S = similar, W = worse.

The results reported in Table 6.9 relate exclusively to the three 200-sentence test sets, each of which contained only sentences for which TransBooster and the baseline MT systems produced a different output. In order to estimate manual evaluation scores on the entire 800-sentence test set, we extrapolated these scores by taking into account the number of sentences for which TransBooster and the baseline systems produced an identical output³ and by scaling the scores in Table 6.9 based on a 200-item test set to the total amount of different sentences as reported in Table 6.8.

	TB vs. LogoMedia			TB vs. Systran			TB vs. SDL		
	B	S	W	B	S	W	B	S	W
Fluency %	14.87	74.88	10.25	12.37	76.13	11.50	18.62	70.63	10.75
Accuracy %	11.12	79.13	9.75	12.00	75.63	12.37	17.37	72.75	9.88

Table 6.10: Extrapolation of the manual evaluation results in Table 6.9 for the entire 800-sentence test set. B = better, S = similar, W = worse

The results in Table 6.10 are an estimate of the manual evaluation on the entire 800-sentence test set. Overall, evaluators considered TransBooster to outperform LogoMedia and SDL both on fluency (14.87% *better* vs. 10.25% *worse* for LogoMedia, 18.62% *better* vs. 10.75% *worse* for SDL) and accuracy (11.12% *better* vs. 9.75% *worse* for LogoMedia, 17.37% *better* vs. 9.88% *worse* for SDL). For Systran, the manual evaluations show a similar proportion of improved/worse translations, both for accuracy as for fluency.

In general, the differences between the *better* and *worse* percentages are slightly larger for fluency than for accuracy. In other words, fluency improves (a little bit) more than accuracy. This could be explained by the fact that the linguistic expert evaluators were

³For LogoMedia: 475 sentences, for Systran: 432 sentences, for SDL: 433 sentences, cf. Table 6.8.

asked to give a *comparative* evaluation of the sentence pairs. While only a single lexical change or a slightly different word order can be sufficient to make a target sentence more fluent, this difference might not be sufficient to make the target sentence semantically more similar to the original. This might even be more so in the case of relatively poor baseline MT output. Given the highly specialised nature of the test sentences⁴, wide-coverage RBMT systems like the ones used in the experiments are not likely to produce an output with a high degree of accuracy and fluency without specific lexical tuning to the subdomain being translated (Hutchins and Somers, 1992).

In the following section, we will analyse the type of phenomena that TransBooster improves on and explain why some sentences receive a translation which is worse than the original baseline output.

6.2.2.3 Analysis

By breaking down a complex input sentence into a number of simpler chunks and spoon-feeding them to the baseline MT system, TransBooster can help a baseline MT system improve its own output. All observed improvements are due to TransBooster’s complexity reduction, which allows the baseline MT system’s analysis, transfer and generation modules to operate at optimal strength.

At the surface level, improvements can be divided into four different classes: (i) better target language lexical selection; (ii) better source language homograph resolution; (iii) improved agreement; (iv) improved word order. The first class of improvements (‘better target language lexical selection’) corresponds to an improved treatment of polysemy: the same source word has different translations depending on the lexico-semantic context that the word was used in. For example, the word ‘wood’ can refer to the substance under the bark of a tree (in Spanish ‘madera’) or to a geographical area with many trees (in Spanish ‘bosque’). The second class of improvements corresponds to a better treatment of homography in the source language. Homographs are different words that happen to share the same spelling. For examples, the word ‘bark’ as the sound of a dog (in Spanish ‘ladrido’) is completely unrelated to the word ‘bark’ as the covering of a tree (in Spanish

⁴All sentences were selected from Section 23 of the Wall Street Journal section of the Penn-II Treebank, which contains material extracted from business-related press articles.

'corteza'). The third class ('correct inflection of the target word') and fourth class ('correct word order') are also due to the reduced complexity of the input chunks.

Although it is difficult to measure the exact weight of each of these four categories on the overall improvements, a manual analysis of the improvements showed that approximately 35% of the improvement was due to better target language lexical selection, 35% to improved word order in target, 20% to better source language homograph resolution and 10% to improved agreement. Table 6.11 contains a number of example sentences that illustrate each of the four above-mentioned improvements.

Original	On days <u>like</u> Friday, that <u>means</u> they must buy shares from sellers when no one else is willing to.
Systran	El días <u>tener gusto de</u> viernes, ese <u>los medios</u> que deben comprar partes de vendedores cuando ningunos otros están dispuestos a
TransBooster	En días <u>como</u> viernes, eso <u>significa</u> que deben comprar partes de vendedores cuando ningunos otros están dispuestos a
Analysis	Homograph resolution: 'like' analysed as preposition (correct 'como') instead of as verb (erroneous 'tener gusto de') + 'means' correctly analysed as verb (correct 'significa') instead of as noun (erroneous 'los medios').
Original	This month, however, Businessland warned investors <u>that</u> results for its first quarter <u>ended</u> Sept. 30 hadn't <u>met</u> expectations.
LogoMedia	Este mes, sin embargo, Businessland advirtió que los inversionistas <u>a quienes</u> los resultados por su primer trimestre <u>terminaron</u> 30 de sep no hubieran <u>cubierto</u> las expectativas.
TransBooster	Este mes, sin embargo, Businessland advirtió a inversionistas <u>que</u> los resultados por su primer trimestre <u>terminado</u> 30 de sep no haban <u>satisfecho</u> las expectativas
Analysis	Lexical selection: TransBooster improves the translation of 'met' by LogoMedia ('cubierto') to the better 'satisfecho' Homograph resolution: 'that' is correctly analysed as a complementiser ('que') instead of as a relative pronoun ('a quienes'). Improved analysis: 'ended' is correctly interpreted as a complement past participle ('terminado') instead of as a main verb ('terminaron').
Original	A Flemish game show has as its host a Belgian <u>pretending</u> to be Italian
SDL	Un programa concurso Flamenco tiene como su anfitrión que un <u> fingir</u> belga ser italiano
TransBooster	Un programa concurso Flamenco tiene como su anfitrión a un belga <u> fingiendo</u> para ser italiano
Analysis	Improved analysis: 'pretending' is correctly inflected in the output produced by TransBooster ('fingiendo') instead of the pure infinitive form ('fingir') produced by LogoMedia. Word order: better word order in output TransBooster
Original	"It's terrific for advertisers to <u>know</u> the reader <u>will be</u> paying more," <u>said</u> Michael Drexler, national media director at Bozell Inc. ad agency.

Continued on next page

LogoMedia	“Es excelente que anunciantes <u>saber</u> que el lector <u>estar</u> pagando mayor cantidad”, Michael Drexler director de medios de comunicación nacional en Bozell Inc <u>dijo</u> Agencia de publicidad
TransBooster	“Es excelente que anunciantes <u>sepan</u> que el lector <u>estará</u> pagando mayor cantidad,” <u>dijo</u> Michael Drexler, director de medios de comunicacin nacional en Bozell Inc. Agen- cia de publicidad
Analysis	Inflection: correct inflection of erroneous ‘saber’ (LogoMedia) → ‘sepan’ (Trans- Booster) and of the erroneous ‘estar’ (LogoMedia) → ‘estará’ (TransBooster). Word Order: better word order in output TransBooster (placement of ‘dijo’ (= ‘said’)) which makes the TransBooster output much more fluent than LogoMedia.
Original	For his sixth novel, Mr Friedman tried to resuscitate the protagonist of his 1972 <u>work</u> , “About Harry Towns ”
LogoMedia	Para su sexta novela, el Sr Friedman trató de resucitar al protagonista de su 1972 <u>trabajo</u> , “Sobre Harry pueblos ”
TransBooster	Para su sexta novela, El Sr Friedman trató de resucitar al protagonista de su 1972 <u>obra</u> , “Sobre Harry pueblos ”
Analysis	Lexical selection ‘work’ is correctly translated as ‘obra’ (‘artistic work’) instead of ‘trabajo’ (‘labour’).

Table 6.11. Examples of each of the four areas of TransBooster improvements. lexical selection, word order, agreement, homograph resolution

Complexity reduction, even when correctly executed, does not necessarily lead to improvements. If the MT systems needs the entire syntactic structure of the original sentence, including adjuncts, to correctly generate the output, or if it relies on certain lexico-semantic information in omitted adjuncts for lexical selection, translations might worsen, as is shown by the examples in Table 6.12.

6.2.2.4 The impact of parser-based input

The results in Section 6.2.2 were obtained by using the 800-sentence Penn-II human parse-annotated sentences. If TransBooster is to be used as a wrapper application on top of an MT system in a real-world application, unseen input will have to be parsed into a Penn-II-like structure in a step previous to the TransBooster decomposition. Obvious candidates for the front-end parsing are current state-of-the art statistical parsers such as (Charniak, 2000) and (Bikel, 2002).⁵ Both parsers employ history-based, generative, lexicalised models and achieve results of almost 90% labelled f-score when tested on the trees in Section 23 of the Penn-II Treebank.

In order to quantify the impact of the use of parsing technology on the advantages

⁵(Bikel, 2002) is a Java implementation emulating (Collins, 1999) Model 2.

Original SDL TransBooster Analysis	A <u>bus</u> is the data highway within a computer Un <u>bus</u> es la autopista de datos dentro de una computadora. Un <u>autobús</u> es la autopista de datos dentro de una computadora. The reduced complexity of the argument skeleton 'A bus is the highway' leads the baseline MT system to translate 'bus' erroneously as 'autobús' instead of the correct 'bus' (= 'busbar in a computer'). SDL needs the presence of 'data highway' or 'computer' to correctly translate 'bus'
Original LogoMedia TransBooster Analysis	One doubter is George Krug, a chemical-industry analyst at Oppenheimer & Co and a <u>bear</u> on plastics stocks Un escéptico es George Krug, un analista químico - industria en Oppenheimer & Co. y un <u>bajista</u> sobre acciones de plásticos Un escéptico es George Krug, un químico - analista industrial en Oppenheimer & Co y un <u>oso</u> sobre acciones de plásticos At a certain point during the TransBooster decomposition, the string 'a bear on plastics stocks' is sent to the baseline MT system for translation. The lack of additional financial vocabulary leads LogoMedia to translate 'bear' literally as 'oso' (= 'bear' as a mammal) instead of the correct 'bajista' (= 'bear' as a type of investor)
Original Systran TransBooster Analysis	In an unusual move, several funds <u>moved</u> to calm investors with recordings on their toll-free phone lines En un movimiento anormal, algunos fondos <u>cambiaban de lugar</u> a la calma inversionistas con grabaciones sobre sus líneas de teléfonos de número gratuito En un movimiento inusual, varios fondos <u>movidos</u> a los inversionistas tranquilos con las grabaciones en sus líneas telefónicas gratis At a certain point during the TransBooster decomposition, the string 'several funds moved to calm investors.' is sent to the baseline MT system for translation. Despite the fact that this is a correct simplification of the original, more complex sentence, Systran translates 'moved' erroneously as a past participle modifier ('movidos') instead of as the main verb of the original sentence ('cambiaban de lugar')

Table 6.12 Examples of sentences in which a correct complexity reduction leads to worse translation.

gained from TransBooster's complexity reduction, we repeated exactly the same experiments as reported in Section 6.2.2.1, but instead of using the human parse-annotated structures of the Penn-II Treebank as input to our algorithm, we used the parser output of (Charniak, 2000) and (Bikel, 2002). Tables 6.13 and 6.14 show the results of this experiment.

When comparing the results in Tables 6.13 and 6.14 to the results in Table 6.3, we observe that the relative performance of TransBooster with respect to the baseline systems drops between 1.3–1.8% BLEU score, 0.7% NIST score and 0.4–0.6% GTM score when using (Charniak, 2000) and between 1.5–1.7% BLEU score, 0.5–0.6% NIST score and 0.6–1.0% GTM score when using (Bikel, 2002).

	BLEU	NIST	GTM
LogoMedia	0.3140	7.3272	0.5627
TransBooster	0.3140	7.3145	0.5632
Percent of Baseline	100.0%	99.8%	100.0%
Systran	0.3003	7.1674	0.5553
TransBooster	0.2987	7.1643	0.5547
Percent of Baseline	99.4%	99.9%	99.9%
SDL	0.3039	7.2735	0.5657
TransBooster	0.3035	7.2974	0.5642
Percent of Baseline	99.9%	100.3%	99.7%

Table 6.13: TransBooster results on 800-sentence test set, parsed with (Charniak, 2000)

	BLEU	NIST	GTM
LogoMedia	0.3140	7.3272	0.5627
TransBooster	0.3141	7.3203	0.5601
Percent of Baseline	100.0%	99.9%	99.5%
Systran	0.3003	7.1674	0.5553
TransBooster	0.2973	7.1720	0.5542
Percent of Baseline	99.0%	100.0%	99.8%
SDL	0.3039	7.2735	0.5657
TransBooster	0.3044	7.3076	0.5620
Percent of Baseline	100.2%	100.5%	99.3%

Table 6.14: TransBooster results on 800-sentence test set, parsed with (Bikel, 2002)

This decrease in performance is caused by the inevitable noise introduced by the use of statistical parsers. Despite f-score figures of both (Charniak, 2000) and (Bikel, 2002) of almost 90%, the mislabelling of one single constituent by the parser can be sufficient to lead to an erroneous TransBooster decomposition, which might cause wrong chunk translations by the baseline systems.

For example, consider the parses of the chunk ‘a Belgian pretending to be Italian’ in Figures 6.1 and 6.2. The selected chunk is part of the evaluation sentence ‘A Flemish game show has as its host a Belgian pretending to be Italian’ in Table 6.11, in which TransBooster’s improvements over the baseline translation by SDL are explained.

Apart from the differences between Figures 6.1 and 6.2, the parser output of (Bikel, 2002) is exactly the same as the human parse-annotated Penn-II version of the whole sentence. On this single sentence, (Bikel, 2002) achieves labelled bracketing precision/recall figures of 80% and 66.67% respectively. If instead of the human-parse annotated version of

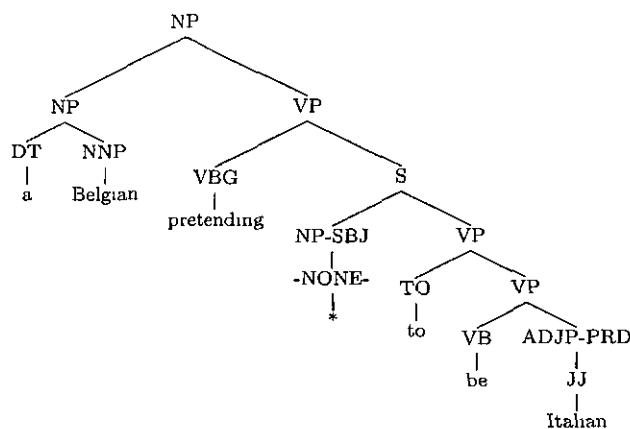


Figure 6 1 The human parse-annotated structure of the chunk ‘a Belgian pretending to be Italian’ in the Penn-II Treebank

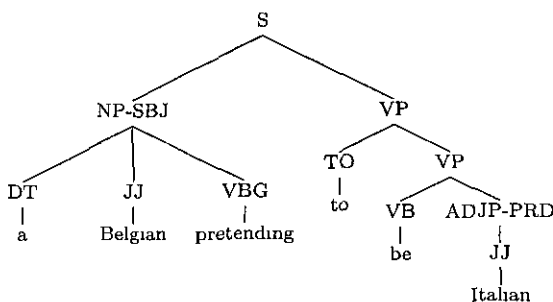


Figure 6 2 The parser output of (Bikel, 2002) of the chunk ‘a Belgian pretending to be Italian’

the sentence, the parser output of (Bikel, 2002) is provided as input into the decomposition algorithm, TransBooster produces the result in (86):

(86) ‘Un programa concurso Flamenco tiene como su anfitrión un fingir belga para ser italiano.’

This time, the result in (86) is not substantially better than the output produced by SDL. The main reason for this is the parser’s erroneous analysis of ‘a Belgian pretending’ as an *NP*, which leads the decomposition algorithm to send the entire chunk to SDL, leading to the nonsensical translation *‘un fingir belga’.

As explained in Section 6.2.2.1, the reference set of human translations contains a slight bias towards the baseline MT systems. Therefore, we decided to repeat the experiment on the same three unbiased 600-sentence test sets of Section 6.2.2.1. Tables 6.15 and 6.16 contain the results of this experiment.

	BLEU	NIST	GTM
LogoMedia	0.2830	6.8555	0.5391
TransBooster	0.2861	6.8602	0.5422
Percent of Baseline	101.1%	100.1%	100.6%
Systran	0.2708	6.7244	0.5368
TransBooster	0.2722	6.7500	0.5385
Percent of Baseline	100.5%	100.4%	100.3%
SDL	0.2823	6.8917	0.5473
TransBooster	0.2848	6.9389	0.5477
Percent of Baseline	100.9%	100.7%	100.0%

Table 6.15: TransBooster results on the three 600-sentence test sets, parsed with (Charniak, 2000)

	BLEU	NIST	GTM
LogoMedia	0.2830	6.8555	0.5391
TransBooster	0.2848	6.8529	0.5379
Percent of Baseline	100.6%	99.9%	99.8%
Systran	0.2708	6.7244	0.5368
TransBooster	0.2696	6.7409	0.5365
Percent of Baseline	99.6%	100.2%	99.9%
SDL	0.2823	6.8917	0.5473
TransBooster	0.2855	6.9527	0.5456
Percent of Baseline	101.1%	100.9%	99.7%

Table 6.16: TransBooster results on the three 600-sentence test sets, parsed with (Bikel, 2002)

When comparing the results in Tables 6.15 and 6.16 to the results in Table 6.4, we observe that the relative performance of TransBooster with respect to the baseline systems drops between 0.9–1.9% BLEU score, 0.4–0.7% NIST score and 0.3–0.4% GTM score when using (Charniak, 2000) and between 1.7–2.1% BLEU score, 0.5–0.8% NIST score and 0.7–1.1% GTM score when using (Bikel, 2002).

Overall, parsing with (Charniak, 2000) gives a slightly better result than parsing with (Bikel, 2002). The results in Table 6.15 show that, when parsing the input with (Charniak, 2000), the advantages achieved by the TransBooster’s complexity reduction are sufficient to outdo the decrease in performance induced by the parser errors.

6.2.3 Experiments with TransBooster Mark II

All the previously reported results in this chapter refer to the main TransBooster architecture (TB_{MarkI}). As explained in Section 5.3 on page 95, an alternative, simplified algorithm (TB_{MarkII}) was implemented, mainly in order to handle split pivots. The main difference between both approaches is that TB_{MarkII} relies solely on string replacements of satellite SVs in target rather than recursively stitching together chunk translations in target, as is the case for TB_{MarkI} .

During development, we noticed that evaluation scores for TB_{MarkII} consistently lagged behind TB_{MarkI} . This was mainly due to two factors: (i) the main advantage of TB_{MarkII} over TB_{MarkI} is that TB_{MarkII} is able to treat split pivots, a phenomenon common in most Germanic languages. Since we perform our experiments on English→Spanish, this advantage is not visible; (ii) the algorithm in TB_{MarkII} does not allow for adjunct constituents to be omitted in the skeletons sent to the baseline MT systems. Therefore, for the language pair English→Spanish, the use of TB_{MarkII} leads to less complexity reduction than TB_{MarkI} .

Although TB_{MarkII} was not developed to the same extent as TB_{MarkI} , we have included the latest automatic evaluation scores of TB_{MarkII} with respect to the three baseline RBMT systems in Table 6.17. As is clear from these results, TB_{MarkII} is not able to outperform the baseline MT systems.

	BLEU	NIST	GTM
LogoMedia	0.3140	7.3272	0.5627
TransBooster	0.3100	7.2862	0.5591
Percent of Baseline	98.7%	99.4%	99.4%
Systran	0.3003	7.1674	0.5553
TransBooster	0.2967	7.1560	0.5548
Percent of Baseline	98.8%	99.8%	99.9%
SDL	0.3039	7.2735	0.5657
TransBooster	0.3021	7.2653	0.5636
Percent of Baseline	99.4%	99.9%	99.6%

Table 6.17: TransBooster Mark II results on the 800-sentence test set.

6.2.4 TransBooster and Rule-based MT: conclusion

In Section 6.2, we have seen that the output produced by TransBooster shares many characteristics of the baseline MT output, but improves on lexical selection, homograph resolution, word order and agreement features. Most of the improvements are triggered by complexity reduction of the input. Most of the cases in which TransBooster causes the deterioration of the original output are due to context distortion.

Of the three baseline RBMT systems used, TransBooster outperforms two systems (SDL and LogoMedia) and achieves similar results compared to the third one (Systran), both in terms of automatic evaluation and of manual evaluation results. One should be careful not to draw definite conclusions about the quality of an MT system based on relative TransBooster scores alone. For example, that fact that TransBooster achieves only comparable results with respect to Systran, while it clearly outperforms the two other RBMT systems, might lead one to conclude that Systran is the better of the three RBMT systems for the language pair used for evaluation. This conclusion is not correct. According to the automatic evaluation scores in Table 6.3 and based on our own experience with the produced MT output, the better system of the three was LogoMedia. The main reason why TransBooster achieved better relative scores vs. LogoMedia than vs. Systran is that most of the development was done based on output produced by LogoMedia.

The complexity reduction offered by TransBooster can only lead to an improved RBMT output if the baseline system possesses a transfer lexicon that contains translation alternatives to account for homography and polysemy phenomena. When such a lexicon is coupled to a shallow analysis module, as is the case for most commercial RBMT systems, TransBooster has the potential to improve the original translation quality.

6.3 Results for Data-driven MT

In Section 6.2, we showed results of TransBooster interfaced with three commercial wide-coverage RBMT systems. This section contains experimental results of TransBooster interfaced with two data-driven MT systems, representing the two most important data-driven MT research paradigms at the moment. SMT and EBMT.

6.3.1 TransBooster and SMT

6.3.1.1 Experimental setup

The baseline MT system for our experiments⁶ was a phrase-based SMT system (English \rightarrow Spanish) that we constructed using the GIZA++ alignment tool (Och and Ney, 2003)⁷, the SRI Language Modeling Toolkit (Stolcke, 2002)⁸ and the Pharaoh decoder (Koehn, 2004)⁹. We used an interpolated tri-gram language model with Kneser-Ney discounting (Kneser and Ney, 1995). Since the SMT system was constructed with the Pharaoh decoder, we will refer to the entire SMT system as PHARAOH in the rest of this section.

The data used to train the system was taken from the English-Spanish section of the Europarl corpus (Koehn, 2005). From this data, 501K sentence pairs were randomly extracted from the designated training section of the corpus and lowercased. Sentence length was limited to a maximum of 40 words for both Spanish and English, with sentence pairs having a maximum relative sentence length ratio of 1.5. From this data we used the method of (Och and Ney, 2003) to extract phrase correspondences from GIZA++ word alignments.

Following this method, word alignment is performed in both source-target and target-source directions. These uni-directional alignments are then combined and the intersection is taken. These highly confident word alignments are then extended by iteratively adding adjacent alignments present in the union of the unidirectional alignments. In a final step, alignments are added that occur in the union, where both the source and target words are unaligned. Source-target phrase pairs can then be extracted based on these alignments, with probabilities estimated from relative frequencies. For our experiments phrase length was limited to 6 words.

For testing purposes two sets of data were used, each consisting of 800 English sentences. The first set was randomly extracted from section 23 of the WSJ section of the Penn-II Treebank; the second set consists of randomly extracted sentences from the test

⁶The experiments in this section were carried out in collaboration with my colleagues K. Owczarzak and D. Groves.

⁷<http://www.fjoch.com/GIZA++.html>

⁸<http://www.speech.sri.com/projects/srilm>

⁹<http://www.isi.edu/licensed-sw/pharaoh/>

section of the Europarl corpus, which had been parsed with (Bikel, 2002).¹⁰

We decided to use two different sets of test data instead of one because we are faced with two ‘out-of-domain’ phenomena that have an influence on the scores, one affecting the TransBooster algorithm, the other the phrase-based SMT system. On the one hand, the TransBooster decomposition algorithm performs better on ‘perfectly’ parse-annotated sentences from the Penn Treebank than on the output produced by a statistical parser such as (Bikel, 2002), which introduces a certain amount of noise. On the other hand, Pharaoh was trained on data from the Europarl corpus, so it performs much better on translating Europarl data than out-of-domain Wall Street Journal text.

Parameter	Value
p_ChunkLength	13
p_PivotLength	4
p_PivotAttach	3
p_PivotCheck	true
p_SatDefault	‘adj’

Table 6 18: Optimal parameter settings for the TransBooster-Pharaoh interface

Table 6 18 contains the optimal parameter settings for the TransBooster-Pharaoh interface. The main difference with the optimal settings in Table 6.2 is the value of p_ChunkLength. For TransBooster-Pharaoh, only chunks containing more than 13 lexical items are subjected to the decomposition process. The fact that the optimal value of p_ChunkLength is 13 for the SMT system compared to 4 and 5 for the RBMT systems¹¹ might reflect the fact that SMT systems are better at handling local phenomena, at constituent level, than at global reordering issues, which require more syntactic knowledge.

¹⁰Contrary to the RBMT experiments reported in section 6.2, we did not use (Charniak, 2000) to parse the input. There are two reasons for this. (i) the goal of this chapter is to evaluate the performance of TransBooster on the main current MT architectures, not to use it as a task-based evaluation platform for parsing technology, (ii) due to the extended average time required for a single TransBooster-Pharaoh run (approximately 60 min for translating 800 sentences), we discarded development with (Charniak, 2000) after initially obtaining better results with (Bikel, 2002).

¹¹p_ChunkLength = 4 (Systran) and p_ChunkLength = 5 (LogoMedia and SDL) gives optimal results for the RBMT systems.

6.3.1.2 Results

Automatic Evaluation Table 6.19 contains a comparison between TransBooster and Pharaoh on the Europarl test set. TransBooster improves on Pharaoh with a statistically significant relative improvement of 3.3% in BLEU and 0.6% in NIST score. Surprisingly, the GTM score obtained by TransBooster is 0.4% lower than Pharaoh’s results. This is most probably due to an issue with punctuation. Contrary to BLEU/NIST, which treat punctuation marks as separate tokens, GTM does not distinguish punctuation marks as separate tokens. Since TransBooster joins the end-of-sentence punctuation mark to the final letter of the output in a post-processing step, this can lead to a number of mismatches in the case of a fully tokenised reference translation and an evaluation metric that does not use tokenisation as a preprocessing step. After removing punctuation in both reference and output translations, we observed a rise of the relative GTM scores from 99.6% to 100.1%

	BLEU	NIST	GTM
Pharaoh	0.1986	5.8393	0.5439
TransBooster	0.2052	5.8766	0.5419
Percent of Baseline	103.3%	100.6%	99.6%

Table 6.19: TransBooster vs Pharaoh Results on the 800-sentence test set of Europarl

For the same reasons mentioned before in Section 6.2.2.2, it is necessary to corroborate these automatic evaluation scores with a manual evaluation, which we will extend on in the following section.

	BLEU	NIST	GTM
Pharaoh	0.1343	5.1432	0.5054
TransBooster	0.1379	5.1259	0.4954
Percent of Baseline	102.7%	99.7%	98%

Table 6.20. TransBooster vs. Pharaoh: Results on the 800-sentence test set of the WSJ

The comparison between TransBooster and Pharaoh on the Wall Street Journal test set is shown in Table 6.20. As with Europarl, TransBooster improves on Pharaoh according

to the BLEU metric, but falls slightly short of Pharaoh's NIST and GTM scores. In contrast to the scores on the Europarl corpus, these results are not statistically significant according to a resampling test (on 2000 resampled test sets) with the toolkit described in Zhang and Vogel (2004)¹² Although the input to TransBooster in this case are near to perfect human parse-annotated sentences, we are not able to report statistically significant improvements over Pharaoh. This can be explained by the fact that the performance of phrase-based SMT systems on out-of-domain text is very poor (items are left untranslated, etc.) as is described in (Koehn, 2005) and indicated by the much lower absolute test scores of Table 6.20 in comparison to table 6.19. In other words, in this case it is more difficult for TransBooster to help the SMT system to improve on its own output through syntactic guidance.

Manual Evaluation With the optimal settings shown in Table 6.18, TransBooster produced a result different from Pharaoh for 185 sentences (= 23.12%) in the 800-sentence Europarl test set. The reason for this high back-off percentage is the fact that the optimal results are produced by only decomposing chunks that dominate 13 or more leaf nodes

The 185 sentences were randomly distributed between the same eight linguistic experts mentioned earlier who were asked to evaluate the sentences following the criteria outlined in Section 6.2.2.2. Table 6.21 contains the results of the manual evaluation. These results were extrapolated to the entire 800-sentence test set by taking into account the 615 sentences for which TransBooster and Pharaoh produced an identical output. The results of this extrapolation are shown in Table 6.22. Overall, evaluators considered TransBooster to outperform Pharaoh both on fluency (10.13% *better* vs. 3.5% *worse*) and accuracy (10.88% *better* vs. 3.0% *worse*).

Surprisingly, when comparing these results in to the results in Tables 6.9 and 6.10, TransBooster seems to perform better when interfaced to an SMT system than to RBMT systems. This can be explained by the fact that the baseline SMT system that we constructed operates without any explicit syntactic knowledge and benefits more from TransBooster's syntactic guidance than RBMT systems. In addition, one should take into account that since PHARAOH is merely a 'vanilla' baseline phrase-based SMT system, its

¹²<http://projectile.is.cs.cmu.edu/research/public/tools/Bootstrap/tutorial.htm>

overall output quality is significantly lower than the output of the RBMT systems, as can be deduced from comparing the SMT scores in Tables 6.19 and 6.20 to the RBMT scores in Table 6.3, which might make it easier to improve on than the better performing RBMT systems.

	TB vs. Pharaoh		
	B	S	W
Fluency	43.8%	41.0%	15.2%
Accuracy	47.0%	40.0%	13.0%

Table 6.21: Comparative results of the manual evaluation of TransBooster vs. Pharaoh. B = better, S = similar, W = worse.

	TB vs Pharaoh		
	B	S	W
Fluency	10.13%	86.37%	3.5%
Accuracy	10.88%	86.12%	3.0%

Table 6.22: Extrapolation of the manual evaluation results in Table 6.21 for the entire 800-sentence test set. B = better, S = similar, W = worse.

In the next section, we analyse the differences between the output translations of Pharaoh and TransBooster, and provide a number of example translations.

6.3.1.3 Analysis

The majority of improvements (70%) by invoking the *TransBooster method on Pharaoh* are caused by a better word order. This is because it is syntactic knowledge and not a linguistically limited language model that guides the placement of the translation of the decomposed input chunks. Moreover, smaller input chunks, as produced by TransBooster and translated in a minimal context, are more likely to receive correct internal ordering from the SMT language model.

The remaining 30% of improvements resulted from a better lexical selection. This is caused not only by shortening the input, but mainly by *TransBooster being able to separate the input sentences at points of least cohesion, namely, at major constituent boundaries*. It is plausible to assume that probability links between the major constituents are weaker than inside them, due to data sparseness, so translating a phrase in the context of only the

Original	Despite an <u>impressive number</u> of international studies , there is still no <u>clear evidence</u> of any direct link between violence and media consumption
Pharaoh	a pesar de los estudios internacionales , todavía no existe ninguna relación directa entre la violencia y media un número impresionante pruebas claras de consumo
TransBooster	pese a un número impresionante de estudios internacionales , todavía no hay pruebas claras de ninguna relación directa entre la violencia y los medios consumo
Analysis	<u>word order</u> , <u>better placement of the translations of 'an impressive number' and 'clear evidence'</u>
Original	The European Union is jointly responsible, with the <u>countries of origin</u> , for immigration and for <u>organising</u> those migration flows, which are so necessary for the development of the region
Pharaoh	la unión europea es corresponsable de inmigración y de los flujos migratorios, que son necesarias para el desarrollo de la región, <u>con los países de origen, organizador.</u>
TransBooster	la unión europea es corresponsable, <u>con los países de origen</u> , de inmigración y de los flujos migratorios, que son necesarias para <u>organizar</u> el desarrollo de la región
Analysis	<u>word order and lexical selection</u> , better placement of the translation of 'with the countries of origin'. In addition, TransBooster translates ' <u>organising</u> ' correctly as a verb ('organizar'), while Pharaoh translates it erroneously as a noun/adjective ('organizador').
Original	Presidency communication on the situation in <u>the Middle East</u>
Pharaoh	presidencia comunicación sobre la situación en <u>el mediterráneo</u>
TransBooster	presidencia comunicación sobre la situación en <u>el cercano oriente</u>
Analysis	<u>lexical selection</u> , improved translation of 'the Middle East' from <u>el mediterráneo</u> ('the mediterranean') to the correct 'el cercano oriente'.
Original	<u>I am proud of the fact</u> that the Committee on Budgetary Control has been able to agree unanimously on a draft opinion within a very short period of time
Pharaoh	me alegra el hecho de que la comisión de presupuestos ha podido dar mi aprobación unánime sobre un proyecto dictamen en un periodo de tiempo muy corto
TransBooster	estoy orgulloso del hecho que la comisión de presupuestos <u>ha llevado a acuerdo unánime</u> sobre un proyecto dictamen en un periodo de tiempo muy corto
Analysis	<u>lexical selection</u> improved translation of 'I am proud of' from the erroneous 'me alegra' ('I am happy about') to the correct 'estoy orgulloso'. In addition, the translation of 'agree unanimously' by Pharaoh agrees with the wrong subject ('I'), instead of the correct 'the Committee', as enforced by TransBooster.

Table 6.23: Examples of improvements over Pharaoh word order and lexical selection.

heads of neighbouring constituents might actually help. Table 6.23 illustrates the main types of improvements with a number of examples.

6.3.2 TransBooster and EBMT

The experiments reported in this section were mainly carried out by my colleagues K. Owczarzak and D. Groves in preparation for (Owczarzak et al., 2006) at the 7th Biennial Conference of the Association for Machine Translation in the Americas. They are included in this dissertation because they are based on the TransBooster technology and complement the SMT experiments in Section 6.3.1 with an insight into the performance of TransBooster on an EBMT baseline system.¹³

6.3.2.1 Marker-based EBMT

The baseline EBMT system used in the experiments is the NCLT’s marker-based MATREX system (Armstrong et al., 2006). Marker-based EBMT is an approach to EBMT which uses a set of closed-class words to segment aligned source and target sentences and to derive an *additional set of lexical and phrasal resources*. This approach is based on the ‘Marker Hypothesis’ (Green, 1979), a universal psycholinguistic constraint which posits that languages are ‘marked’ for syntactic structure at surface level by a closed set of specific lexemes and morphemes. In a preprocessing stage, the source–target aligned sentences are segmented at each new occurrence of a marker word (e.g. determiners, quantifiers, conjunctions etc).

In order to describe this resource creation in more detail, consider the English–Spanish example in (87):

- (87) ‘You click on the red button to view the effect of the selection.’ → ‘Usted cliquea en el botón rojo para ver el efecto de la selección ’

The first stage involves automatically tagging each closed-class word in (87) with its marker tag, as in (88):

- (88) ‘<PRON> You click <PREP> on <DET> the red button <PREP> to view <DET> the effect <PREP> of <DET> the selection ’ → ‘<PRON> Usted cliquea <PREP> en <DET> el botón rojo <PREP> para ver <DET> el efecto <PREP> de <DET> la selección.’

Taking into account marker tag information (label, and relative sentence position), and

¹³My direct contributions to this section are: (i) the development of the TransBooster application, (ii) a contribution to the development of the EBMT baseline system, and (iii) the analysis of the results.

lexical similarity (via mutual information), the marker chunks in (5) are automatically generated from the marker-tagged strings in (88):

- (89) a. You click <PREP> : <PRON> Usted cliquea
b. <PREP> on the red button <PREP> en el botón rojo
c. <PREP> to view : <PREP> para ver
d. <DET> the effect <DET> el efecto
e. <PREP> of the selection <PREP> de la selección

The marker set used in the experiments consisted of determiners, prepositions, conjunctions, personal pronouns, possessive pronouns, quantifiers and wh-adverbs, following (Gough and Way, 2004; Gough, 2005).

6.3.2.2 Experimental setup

The baseline EBMT system made use of the Marker-Based methods described in Section 6.3.2.1 to extract the chunk-level lexicon. For English, information from the CELEX database¹⁴ was used to create a list of marker words used during segmentation and alignment. The marker word list for Spanish was created by merging two stop-word lists generously supplied by colleagues at the Polytechnic University of Catalunya (UPC) and the University of Barcelona (UB).

After chunking, the resulting source and target marker chunks were aligned using a best-first dynamic programming algorithm, employing chunk position, word probability, marker tag and cognate information to determine sub-sentential links between sentence pairs.

In addition to these chunk alignments, statistical techniques were used to extract a high quality word-level lexicon (which in turn was used during the chunk alignment process). Following the refined alignment method of Och and Ney (2003), the GIZA++ statistical word alignment tool was used to perform source-target and target-source word alignment. The resulting ‘refined’ word alignment set was then passed along with the chunk database to the same system decoder as was used for the SMT experiments (Pharaoh, (Koehn, 2004)). Since Pharaoh was used as the decoder, the MaTrEx system is more an ‘example-

¹⁴<http://www.ru.nl/celex/>

based SMT system' (in terms of the terminology of (Groves and Way, 2005, 2006)) than a 'pure' EBMT system as in (Gough and Way, 2004; Gough, 2005).

The EBMT system was trained on a subsection of the English–Spanish section of the Europarl Corpus. The corpus was filtered based on sentence length (maximum sentence length set at 40 words for Spanish and English) and relative sentence length ratio (a relative sentence length ratio of 1.5 was used), resulting in 958K English–Spanish sentence pairs.

The experiments reported in the next section are based the same testing procedure as the one employed for the SMT experiments, as we explained in Section 6.3.1.1 on page 118. Two test sets were used, each consisting of 800 English sentences. The first set was randomly extracted from Section 23 of of the WSJ section of the Penn-II Treebank. The second set contained randomly extracted sentences from the test section of the Europarl corpus, previously parsed with (Bikel, 2002). The reason for using two different test sets for the EBMT experiments is to account for the same two 'out-of-domain' phenomena that we explained in Section 6.3.1.1.

6.3.2.3 Results

Automatic Evaluation Tables 6.24 and 6.25 contain the automatic evaluation results of TransBooster vs. the EBMT system on the Europarl and test sets respectively. The evaluation was conducted after removing punctuation from the reference and translated texts, and, in the case of the Europarl test set, after removing 59 sentences containing hyphenated compounds that were incorrectly parsed by (Bikel, 2002), thereby omitting a number of sentence-level errors introduced by the parser which could have a negative impact on the TransBooster scores.

On the Europarl test set, TransBooster improves on the EBMT baseline system with 1.0% relative BLEU score and 0.2% relative NIST score. On the WSJ test set, TransBooster achieves relative improvements of 3.8% BLEU score and 0.5% NIST score.

Manual Evaluation In order to corroborate the automatic evaluation scores, 100 sentences were randomly extracted from the Europarl test set. Their baseline translation was compared with that assisted by TransBooster by a human judge with near-native Span-

	BLEU	NIST
EBMT	0.2111	5.9243
TransBooster	0.2134	5.9342
Percent of Baseline	101%	100.2%

Table 6.24: TransBooster vs. EBMT. Results on the 800-sentence test set of Europarl

	BLEU	NIST
EBMT	0.1098	4.9081
TransBooster	0.1140	4.9321
Percent of Baseline	103.8%	100.5%

Table 6.25: TransBooster vs. EBMT. Results on the 800-sentence test set of the WSJ

ish proficiency according to the same manual evaluation guidelines used throughout this dissertation and explained in Section 3.4.1.5. According to the evaluation, out of the 100 sentences, TransBooster improved the fluency of the translation in 55% of the cases, and the accuracy of translation in 53% of the cases.

6.3.2.4 Analysis

Many of the improvements by TransBooster are caused by a better word order in target. Similarly to what we saw in the evaluation on the Pharaoh baseline SMT system in Section 6.3.1.3, the syntactic guidance of TransBooster helps the baseline EBMT system to overcome some of its syntactic limitations.

The other main factor contributing to TransBooster’s improvements is a better lexical selection by the baseline MT system. This can be explained by the fact that the matching procedure of the baseline EBMT system works better when it operates on the previously chunked input presented by TransBooster than when it is confronted with long input strings which are more likely to be wrongly segmented by the baseline system. In other words, TransBooster does an important part of input segmentation for the EBMT system and makes sure that the translated chunks are assembled correctly. Table 6.26 illustrates the main types of improvements with a number of examples.

Original	women have decided that they wish to work, that they wish to make their work compatible with their family life
EBMT	hemos decidido su deseo de trabajar, su deseo de hacer su trabajo compatible con su vida familiar, empresarias
TransBooster	mujeres han decidido su deseo de trabajar, su deseo de hacer su trabajo compatible con su vida familiar
Analysis	word order and lexical selection: The EBMT system translates 'women' erroneously as 'empresarias' ('business women') and inserts this translation at the end of the sentence, giving rise to a wrong word order. 'have decided' is wrongly translated as 'hemos decidido' ('we have decided'). By contrast, the entire constituent 'women have decided' is correctly translated as 'mujeres han decidido' by TransBooster.
Original	if this global warming continues, then part of the territory of the eu member states will become sea or desert
EBMT	si esto continúa calentamiento global, tanto dentro del territorio de los estados miembros tendrán tornarse altamar o desértico
TransBooster	si esto calentamiento global perdurará, entonces parte del territorio de los estado miembros de la unión europea tendrán tornarse altamar or desértico
Analysis	word order and lexical selection: both translations of 'continues' ('continúa' by EBMT, perdurará by TransBooster) are equivalent. However, the location of 'perdurará' in the output is better than 'continúa'. 'part of' is erroneously translated as 'dentro del' ('in') by the EBMT system, while it is correctly translated by TransBooster as 'parte del'. Finally, the EBMT system omits the fact that the states are members of the EU, while TransBooster correctly translates the modifier 'eu' as 'de la unión europea'.
Original	i have noted your appeals and invitations to press ahead , to take advantage of the momentum generated and carry it to nice and beyond
EBMT	he recogido su apelaciones y invitaciones de seguir adelante a que hagan uso de la impulso generado y llevar a animoso y se allende
TransBooster	he recogido sus apelaciones y invitaciones de seguir adelante a que hagan uso de la impulso generado y llevar esto a animoso y más allá
Analysis	agreement and lexical selection: the agreement between the possessive 'your' and the noun 'appeals' improves from the erroneous 'su' produced by the EBMT system to the correct 'sus' of TransBooster. The pronoun 'it' is translated in the TransBooster output ('esto') but is omitted by the EBMT system. 'beyond' is correctly translated as 'más allá' by TransBooster, while the EBMT system produces a nonsensical word sequence ('se allende')

Table 6.26: Examples of improvements over the EBMT baseline: word order and lexical selection.

6.4 Summary

In this chapter, we have analysed the experimental results of TransBooster interfaced with three commercial rule-based systems and two data-driven systems.

For the parse-annotated Penn-II 800-sentence test set, both automatic evaluation and manual evaluation show that TransBooster outperforms two of the three RBMT systems

(SDL and LogoMedia) and achieves similar results compared to the third system (Systran). When parsing the test set with (Charniak, 2000) and (Bikel, 2002), performance drops slightly, as expected, but the gains made by TransBooster’s complexity reduction are strong enough to resist the noise introduced by (Charniak, 2000) when evaluated on the unbiased 600-sentence test sets. The complexity reduction leads the baseline systems to improve on lexical selection (35%), word order (35%), homograph resolution (20%) and agreement (10%)

When interfaced with a phrase-based SMT system, both automatic and manual evaluation scores on a 800-sentence test set extracted from the Europarl corpus clearly show that TransBooster outperforms the SMT system. The additional syntactic guidance of TransBooster leads the SMT system to improve on both word order (70%) and lexical selection (30%). Similar improvements can be seen when TransBooster is interfaced with a marker-based EBMT baseline system.

Overall, both automatic evaluation scores as manual evaluation results seem to indicate that data-driven MT benefits more from the TransBooster technology than RBMT. There are two possible explanations for this: (i) data-driven MT systems benefit more from TransBooster’s syntactic guidance than rule-based systems, and (ii) the baseline data-driven systems were possibly easier to improve on than the more performant rule-based systems used in the experiments.

The results presented in this chapter quantify the effect that TransBooster has on various *single* baseline MT systems. In the next chapter, we will investigate whether it is possible to adapt the TransBooster algorithm so it can take advantage of the combined strength of *multiple* MT systems simultaneously.

Chapter 7

TransBooster as an MEMT interface

7.1 Introduction

In this chapter, we present a novel approach to combining the outputs of multiple MT engines into a consensus translation. In contrast to previous Multi-Engine Machine Translation (MEMT) techniques, we do not rely on word alignments of output hypotheses, but prepare the input sentence for multi-engine processing. We do this by using TransBooster’s recursive decomposition algorithm to produce simple chunks as input to the MT engines. A consensus translation is produced by combining the best chunk translations, selected through majority voting, a trigram language model score and a confidence score assigned to each MT engine.

The chapter is organised as follows: in Section 7.2, we provide a brief introduction to MEMT and present an overview of the most relevant current MEMT techniques. We explain our approach in Section 7.3 and demonstrate it with a worked example. Section 7.4 contains the description, results and analysis of our experiments. Finally, we summarise our findings in Section 7.5.

When comparing the behaviour of TransBooster as an MEMT interface to TransBooster as a wrapper technology on top of an individual MT engine, we will use TB_{MEI} (TransBooster as an MEMT interface) when referring to the former and TB_{SEI} (Trans-

Booster as a single engine interface) when referring to the latter, for purposes of simplicity.

7.2 Multi-engine Machine Translation

7.2.1 Introduction

Multi-Engine Machine Translation (MEMT) is an approach in which multiple MT systems are used simultaneously to produce a consensus translation for the same input text. The assumption underlying MEMT is that the errors committed by one system are independent of the errors committed by other systems. Therefore, by using smart combination techniques on the different MT outputs, it should be possible to select the best parts of each MT system and produce an output which is at least as good as the best of the individual MT outputs.

MEMT is a term coined by Frederking and Nirenburg (1994), who were the first to apply the idea of a multi-engine approach in Natural Language Processing to MT. Researchers in other areas of language technology such as Speech Recognition (Fiscus, 1997), Text Categorisation (Larkey and Croft, 1996) and POS Tagging (Roth and Zelenko, 1998) have also experimented with multi-system approaches. Since then, several researchers in the MT community have come up with different techniques to calculate consensus translations from multiple MT engines, the most important of which are further explained in Section 7.2.2.

An important difference between the multi-engine approach for clear classification tasks such as POS tagging or Text Categorisation and MT is that, in MEMT, the unit for comparison between the different engines is not given *a priori*. Therefore, a crucial step in all previously proposed MEMT techniques is the inferring of the units for comparison by aligning the outputs of the different MT systems. All previous MEMT approaches share one important characteristic they translate the entire input sentence *as is* and operate on the resulting target language sentences to calculate a consensus output. Their main difference lies in the method they use to compute word alignments between the multiple output sentences.

The use of TransBooster as an interface to MEMT is based on a different idea: the

decomposition of each input sentence into optimal chunks by TransBooster can equally be considered as the inferring of the units of comparison for MEMT. In other words, the main novelty of this approach resides in the fact that, in contrast to previous MEMT techniques, we do not rely on word alignments of output hypotheses, but prepare the input sentence directly for multi-engine processing.

7.2.2 Previous Approaches to MEMT

The first MEMT system was produced by Frederking and Nirenburg (1994). They combined the output sentences of three different MT engines, all developed in house: (i) a knowledge-based MT (KBMT) system, the mainline PANGLOSS engine (Frederking et al., 1993), (ii) an example-based MT (EBMT) system (Nirenburg et al., 1993) and (iii) a simple lexical transfer MT system, based on some of the PANGLOSS modules and extended with a machine-readable dictionary (Collins Spanish→English) and a number of other resources. In order to calculate a consensus translation, the authors rely on their knowledge of the inner workings of the engines. They collect sub-sentential chunks of all three engines in a chart data structure, use internal KBMT and EBMT scores¹ to assign a value to each of the chunks and employ a recursive divide-and-conquer procedure to produce the optimal combination of the available chunks by exhaustively comparing all possible combinations of the available chunks. The results of this MEMT system were used in a translator's workstation (TWS) (Cohen et al., 1993), through which a translator either approved the system's output or modified it.

Since the MEMT design of (Frederking and Nirenburg, 1994) is based on the specific internal structure of each of the component engines, the scoring mechanism would have to be redesigned if a new MT engine were to be added. In (Nomoto, 2004), by contrast, the MT engines are treated as black boxes. A number of statistical confidence models are used to select the best output string at sentence level. The confidence models Nomoto (2004) proposes come in two varieties: fluency-based language models (FLMs), which rely on the likelihood of a translation hypothesis in the target language, and alignment-based models (ALMs), which use the IBM translation models (Brown et al., 1993), that measure

¹Until the publication of (Brown, 1996), the quality of the EBMT system was so poor that it hardly ever contributed to the PANGLOSS MEMT engine

how faithful a translation is to its source text. A confidence score indicating the reliability of each individual engine is introduced by biasing the FLMs and ALMs through Support Vector Regression, modifying the scores produced by the language models in such a way that they more accurately reflect the result of an automatic evaluation of the MT systems on a test corpus.

Contrary to (Frederking and Nirenburg, 1994) and (Nomoto, 2004), all other approaches to MEMT rely on word alignment techniques in the translation hypotheses to infer the units for comparison between the MT systems. Bangalore et al. (2001) produce alignments between the different MT hypotheses using ‘progressive multiple alignment’, a popular heuristic solution to multiple alignment in biological sequencing literature (Feng and Doolittle, 1987) based on edit distance (Levenshtein, 1965). For example, the five different MT outputs in Figure 7.1 are aligned into a lattice structure as represented in Figure 7.2.² For each aligned unit, a winner is calculated by selecting the majority translation, or, in cases where there are segments without a clear majority, by using an n -gram language model based on a 58,000 sentence corpus.

English	‘give me driving directions please to middletown area ’
MT1	‘deme direcciones impulsoras por favor a área de middletown ’
MT2	‘deme direcciones por favor a área ’
MT3	‘deme direcciones conductores por favor al área middletown.’
MT4	‘deme las direcciones que conducen satisfacen al área de middletown.’
MT5	‘deme que las direcciones tend en cia a gradan al área de middletown ’

Figure 7.1: An example English sentence and its translation from five different MT systems, from (Bangalore et al , 2001)

The model used by Bangalore et al. (2001) relies on edit distance, which only focuses on insertions, deletions and substitutions. Therefore, this model is not able to correctly align translation hypotheses with a significantly different word order. Jayaraman and Lavie (2005) try to overcome this problem by introducing a more versatile word alignment algorithm that can deal with non-monotone alignments. Alignments in their approach are

²These examples were adapted from (Bangalore et al , 2001)

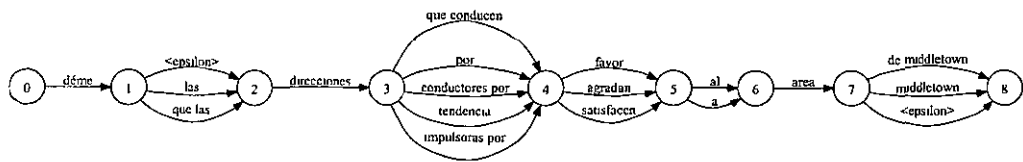


Figure 7.2. Lattice representation of the example sentence in Figure 7 1, from (Bangalore et al , 2001)

produced based on explicit word matches (including morphological variants of the same word and ignoring case) between the various hypotheses, even if the relative location of these matches in the respective hypotheses is very different. A consensus from the MT outputs is calculated by a decoding algorithm that uses the produced alignments, a trigram language model and a confidence score specific to each MT engine.

Another approach to produce a consensus translation from different MT systems was developed by van Zaanen and Somers (2005). Their system, named DEMOCRAT, is a ‘plug-and-play’ MEMT architecture that relies solely on a simple edit distance-based alignment of the translation hypotheses and does not use additional heuristics to compute the consensus translation. DEMOCRAT employs an alignment method similar to the one used by Bangalore et al (2001), but van Zaanen and Somers (2005) explicitly avoid the use of language models or other heuristics that need previous training to ensure that the outputs of different MT engines for all languages can be immediately plugged into their system. DEMOCRAT does not always outperform the best individual MT system, but its ‘plug-and-play’ characteristics make it an option for general users who cannot make up their mind as to which MT system to use and are aiming for a workable ‘average’ translation.

A different way to align translation hypotheses is to use well-established SMT alignment techniques, as in (Matusov et al., 2006), where pairwise word alignments in an entire corpus are used instead of sentence-level alignments. The approach used is similar to the ROVER approach of Fiscus (1997) for combining speech recognition hypotheses. Matusov et al. (2006) consider all possible alignments by iteratively selecting each of the hypothesis translations as a ‘correct’ one and align all other translations with respect to this ‘correct’ hypothesis. The actual alignment is performed in analogy to the training procedure in

SMT, the main difference being that the two sentences that have to be aligned are in the same language. The probabilities of word alignments are calculated based on a test corpus of translations generated by each of the systems. Therefore, the decision on how to align two translations of a sentence takes the whole document context into account. From the obtained alignments, the authors construct a confusion network similar to the approach of Bangalore et al (2001), and derive the best consensus hypothesis by using global system probabilities and other statistical models.

7.3 TransBooster as an MEMT interface

All the MEMT approaches explained in the previous section tackle the problem of how to select or combine the outputs of various MT systems in different ways, but all conclude that combining the outputs, in most cases, results in a better translation than any of the individual contributing outputs. As Frederking and Nirenburg (1994) put it: *‘Three [or more] heads are better than one’*. To date, to the best of our knowledge, all previous MEMT proposals that seek to produce a consensus between several MT outputs operate on MT output for *complete* input sentences.

In the research presented in this chapter, we pursue a different approach: we use the TransBooster decomposition algorithm to split the input string into syntactically meaningful chunks, select the optimal chunk translation from a collection of three MT systems using a number of simple heuristics and rely on TransBooster to recompose the translated chunks in output. Therefore, in contrast to most previous MEMT approaches, the technique we present does not rely on word alignments of target language sentences, but prepares the input sentence for multi-engine processing on the input side.

7.3.1 Algorithm: Overview

Given N different MT engines ($E_1 \dots E_N$), the proposed method recursively decomposes an input sentence S into M syntactically meaningful chunks $C_1 \dots C_M$. Each chunk C_i ($1 \leq i \leq M$) is embedded in a minimal necessary context and translated by all MT engines. For each chunk C_i , the translated output candidates $C_i^1 - C_i^N$ are retrieved and a winner C_i^{best} is calculated based on (i) majority voting, (ii) a language model trained on

a large target language corpus and (iii) a confidence score assigned to each MT engine. In a final step, the output sentence S' is composed by assembling all C_i^{best} ($1 \leq i \leq M$) in their correct target position. A flow chart representing the entire MEMT architecture can be found in Figure 7.3.

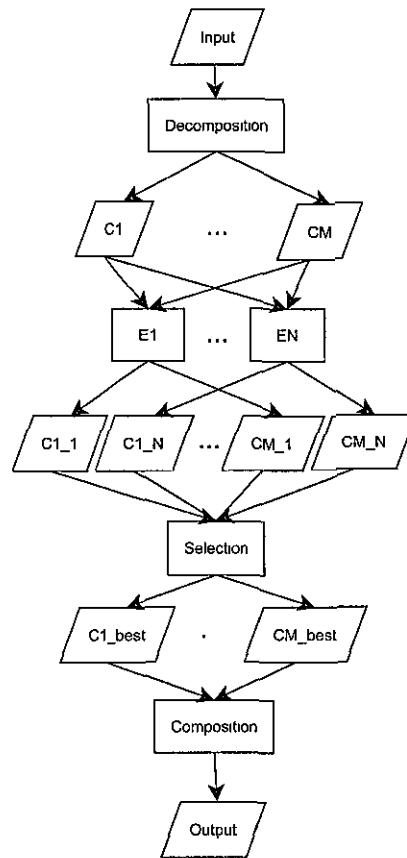


Figure 7.3: A flow chart of the entire MEMT system, with C_i the i^{th} input chunk ($1 \leq i \leq M$), E_j the j^{th} MT engine ($1 \leq j \leq N$) and $C_{i,j}$ the translation of C_i by E_j .

The decomposition into chunks, the tracking of the output chunks in target and the final composition of the output are based on the TransBooster architecture as explained in Chapters 4 and 5.

7.3.2 Algorithm: Details

The algorithm consists of three major parts: (i) decomposition, (ii) selection, and (iii) composition.

In the first part (*'decomposition'*), parallel to what was explained in Chapters 4 and 5, TransBooster decomposes the input S into a number of optimal chunks, embeds these chunks into a context and sends them for translation to each of the N different MT engines ($E_1 \dots E_N$). As before, the input into the algorithm is a Penn Treebank-like syntactic analysis of the input sentence S . In Section 7.4, we report experiments on human parse-annotated sentences (the Penn-II Treebank) and on the output of two state-of-the-art statistical parsers (Charniak, 2000; Bikel, 2002).

In the second part (*'selection'*), the best translation C_i^{best} for each input chunk C_i is selected based on the following three heuristics: (i) majority voting, (ii) a language model trained on a large target language corpus, and (iii) a confidence score assigned to each MT engine.

- 1 **Majority Voting.** Since identical translations by different MT systems are a good indicator of the relative quality of the candidate translations $C_i^1 - C_i^N$, the translation that was produced by the highest number of MT engines is considered to be the best. For example, in the case of MEMT with 5 different MT systems ($MT_1 - MT_5$), if the list of produced translations for chunk C_i is $\{C_i^1 = \text{'a'}, C_i^2 = \text{'b'}, C_i^3 = \text{'c'}, C_i^4 = \text{'a'}, C_i^5 = \text{'d'}\}$, then the output string 'a' is selected as the best translation since it was produced by two MT systems (MT_1 and MT_4), while the other systems produced the mutually distinct translations C_i^2 , C_i^3 and C_i^5 . If no winner is found at this stage, i.e. if the highest number of identical translations is not unique, the second heuristic (*Language Model Score*) is used to select the best translation between the remaining candidates.
- 2 **Language Model Score.** For each produced chunk translation, a Language Model score is assigned by a standard trigram language model trained on 177M words of target language text, comprising the entire training section of the Spanish Europarl Corpus (131M words) (Koehn, 2005), augmented with a corpus of the Spanish newspaper *'La Vanguardia'*³ (46M words). This score is an approximation of the likelihood of the hypothesis translation in the target language and therefore rewards fluency. The Language Model was trained with modified Kneser-Ney smoothing

³<http://www.vanguardia.es>

(Kneser and Ney, 1995) using the SRI Language Modeling Toolkit (Stolcke, 2002).

In the case where Majority Voting produces more than 1 candidate translation, the translation among the selected candidates with the best language model score is considered to be the best. For example, in the case of MEMT with 5 different MT systems ($MT_1 - MT_5$), if the outcome of the Majority Voting procedure leads to $C_i^1 = C_i^4$ and $C_i^2 = C_i^5$, the translation with the highest Language Model score will be selected as the best translation.

3. **Confidence Score.** In the rare cases that no winner is found by either of the previous two heuristics, the best translation is the one produced by the MT engine that obtained the highest BLEU score on the entire test corpus. In the experiments reported in this chapter, this system is LogoMedia (cf. Table 7.4 in Section 7.4.2).

The relative contribution of each of the three above-mentioned heuristics to the MEMT output will be explained during the discussion of the experimental results in Section 7.4.2

In the third part (*'composition'*), the best translations C_i^{best} for each input chunk C_i found by one of the three previously mentioned heuristics, are combined to form the output translation S' . The composition process is essentially the same as explained in Chapters 4 and 5, namely by recursively substituting the retrieved translation of the constituents for the translated SVs in the skeletons. However, since we are operating with multiple MT engines simultaneously, two additional constraints have to be taken into account:

- 1 In case the baseline MT engines use a different reordering of SVs in a particular skeleton, we select the reordering of the MT engine that obtained the highest BLEU score on the entire test corpus (in our case, LogoMedia).
2. If safety measures (cf. Section 5.2.7) demand that a particular MT engine back off from decomposing a chunk and translate the entire chunk as is, then the other MT engines will also operate on the level of the same chunk, even if further decomposition is allowed by them. In other words, the overall granularity of the decomposition, for each chunk, is limited by the MT engine with the lowest degree of granularity. For example, if chunk C_i is decomposed into a pivot and satellites during decomposition, but the safety measures for baseline MT engine E_j ($1 \leq j \leq N$) do not allow it to

carry out this decomposition (e.g. one of the SV translations is not found in the skeleton translated by E_j), then chunk C_i will be the highest level of granularity for *all* remaining MT engines ($E_1 \dots E_{j-1}, E_{j+1} \dots E_N$), even if further decomposition is allowed by them.

7.3.3 A Worked Example

In this section, we will illustrate the use of TransBooster as an MEMT interface to the three baseline RBMT engines that we have been using throughout this dissertation (LogoMedia, Systran and SDL) on example sentence (20) from Section 4.2 on page 40. The output of the example sentence by the baseline systems is displayed in Figure 7.4.

Original	‘The chairman, a long-time rival of Bill Gates, <u>likes fast</u> and confidential deals.’
LogoMedia	‘Al presidente, un rival de mucho tiempo de Bill Gates, <u>les gustan los los</u> tratos rpidos y confidenciales.’
Systran	‘El presidente, rival de largo plazo de Bill Gates, <u>gustos ayuna</u> y los repartos confidenciales.’
SDL	‘El presidente, un rival antiguo de Bill Gates, quiere los tratos rápidos y confidenciales.’

Figure 7 4: Output of example sentence (20) by the three baseline MT engines: LogoMedia, Systran and SDL

The major problems in the translation by LogoMedia are: (i) the wrong number of the pronoun ‘les’ (correct is ‘le’), and (ii) the duplication of the article ‘los’. Systran erroneously analyses the verb ‘likes’ as a noun (\rightarrow ‘gustos’) and identifies the adjective ‘fast’ wrongly as a verb (\rightarrow ‘ayuna’), which renders the output unintelligible. The translation of SDL, by contrast, is acceptable. In what follows, we will explain how TransBooster acts as an MEMT interface, composing selected chunk translations of the individual systems to form the output.

The parse tree of the example sentence in Figure 4.2 on page 41 is used as input to the decomposition module. In a first step, the pivot, arguments and adjuncts are calculated, as in (90):

(90) [The chairman, a long-time rival of Bill Gates,]_{ARG1} [likes]_{pivot} [fast and confidential deals]_{ARG2}

In a second step, the arguments are replaced by syntactically simpler SVs, as in (91):

(91) [The chairman]_{SV_{ARG1}} [likes]_{pivot} [deals]_{SV_{ARG2}}.

The resulting string is translated by each of the three baseline MT engines. For example, the translation produced by Systran is that in (92):

(92) El presidente tiene gusto de repartos.

As explained in previous chapters, this translation allows us (i) to extract the translation of the pivot, and (ii) to determine the location of the translated arguments. This is possible because we determine the translations of the Substitution Variables ('the chairman', 'deals') at runtime. If these translations are not found in (92), we replace the arguments by previously defined SSVs. For example, in (90), we replace 'The chairman, a long-time rival of Bill Gates' by 'The man' and 'fast and confidential deals' by 'cars'. In case the translations of the SSVs are not found (92), we interrupt the decomposition and have the entire input string (20) translated by the MT engine.

We now apply the procedure recursively to the identified chunks 'The chairman, a long-time rival of Bill Gates' and 'fast and confidential deals'.

Since the chunk 'fast and confidential deals' contains fewer words than a previously set threshold,⁴ it is considered ready to be translated by the MT engines. As explained in Section 5.2.5, the chunk has to be embedded in an appropriate context. Again, we can determine the context dynamically ('The chairman likes') or use a static predefined context template ('The man is eating'), mimicking a *direct object context for an NP*.⁵

(93) shows how the chunk 'fast and confidential deals' is embedded in a Dynamic Context.

(93) [The chairman likes]_{DynamicContext} [fast and confidential deals]_{ARG2}

This string is sent to the MT engines for translation. For example, the translation produced by Systran is (94):

⁴All MEMT experiments were performed with `p.ChunkLength = 5`. Cf. Section 6.2.2 for more information

⁵Cf. Appendix E for more detailed information

(94) El presidente tiene gusto de repartos rápidos y confidenciales.

Like DSVs, the translations of Dynamic Contexts are determined at run-time. If we find the translation of the Dynamic Context in (94), it is easy to deduce the translation of the chunk ‘fast and confidential deals’. If, on the contrary, the translation of the Dynamic Context is not found in (94), we back off to a previously defined Static Context template (e.g. ‘The man sees’). In case the translation of this context is not found either, we back off to translating the input chunk ‘fast and confidential deals’ without context.

Since the remaining chunk ‘The chairman, a long-time rival of Bill Gates’ contains more words than the previously set threshold⁴, it is judged too complex for direct translation. The decomposition and translation procedure is now recursively applied to this chunk: it is decomposed into smaller chunks, which may or may not be suited for direct translation, and so forth.

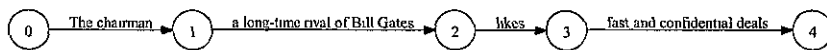


Figure 7.5: Decomposition of Input.

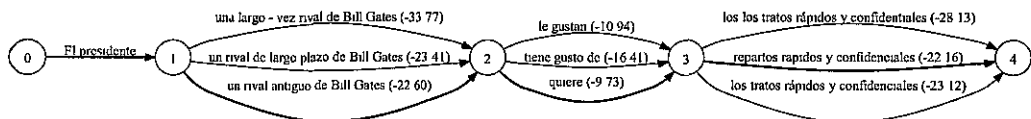


Figure 7.6 Selection of best output chunk. The optimal combination follows the arcs in bold.

The recursive decomposition algorithm splits the initial input string into a number of optimal chunks, which are translated by all MT engines as described above. A simple graph representation of the full decomposition of the input sentence is shown in Figure 7.5. The recovered translations with logprob language model scores are shown in Figure 7.6. From these, the best translations (in bold) are selected as described in Section 7.3.2.

The MEMT combination in Table 7.1 outperforms the outputs produced by Systran and LogoMedia and is similar in quality to the output produced by SDL. Note that our approach is not limited to a blind combination of previously produced output chunks. In

Original	The chairman, a long-time rival of Bill Gates, <u>likes fast</u> and confidential deals
LogoMedia	Al presidente, un rival de mucho tiempo de Bill Gates, <u>les gustan los los</u> tratos rápidos y confidenciales
Systran	El presidente, rival de largo plazo de Bill Gates, <u>gustos ayuna</u> y los repartos confidenciales.
SDL	El presidente, un rival antiguo de Bill Gates, quiere los tratos rápidos y confidenciales.
MEMT	El presidente, un rival antiguo de Bill Gates, quiere repartos rápidos y confidenciales

Table 7.1: Example sentence (20): result of TB_{MEI} vs. baseline MT engines.

the case of Systran, the complexity reduction of the input leads the system to improve on its own translation. In the complete translation (Table 7.1), Systran erroneously analyses the verb ‘likes’ as a noun (\rightarrow ‘gustos’) and identifies the adjective ‘fast’ as a verb (\rightarrow ‘ayuna’). By contrast, examples (93) and (94) show that submitting the chunk ‘fast and confidential deals’ in a simplified context improves the translation of the adjective ‘fast’ from the erroneous ‘ayuna’ in the original translation of the entire sentence by Systran to the correct ‘rápidos’. Also, the translation of the verb ‘likes’ improves to ‘tiene gustos de’, which can only contribute to a better overall MEMT score.

Tables 7.2 and 7.3 contain two more examples that show the benefits of our approach.

Original	‘Imperial Corp, <u>based</u> in San Diego, <u>is</u> the parent of Imperial Savings & Loan.’
LogoMedia	‘Imperial Corp., <u>Fundar</u> en San Diego, <u>ser</u> el padre de Savings & Loan imperial ’
Systran	‘Imperial Corp, basada en San Diego, es el padre de ahorros imperiales y del préstamo.’
SDL	‘Imperial S.a, basado en San Diego, es el padre de Ahorros Imperiales & el Préstamo.’
TB_{MEI}	Imperial Corp., basada en San Diego, es el padre de Savings & Loan imperial.

Table 7.2: Result of TB_{MEI} vs. baseline MT engines on the example sentence ‘Imperial Corp , based in San Diego, is the parent of Imperial Savings & Loan.’

In Table 7.2 the major problems in the translation by LogoMedia are: (i) ‘based’ is erroneously translated as ‘Fundar’ = ‘to found’, and (ii) ‘ser’ = ‘to be’ is not conjugated. Both Systran and SDL correctly conjugate the verb ‘ser’ \rightarrow ‘es’ and select the correct

verb lemma ‘basar’ as translation of ‘based’. However, instead of leaving the proper name (‘Imperial Savings & Loan’) untranslated, as in the case of LogoMedia, they translate each word composing the name separately, which results in which results in awkward results (‘ahorros imperiales y del préstamo’ and ‘Ahorros Imperiales & el Préstamo’ respectively). The MEMT output improves on each of the baseline systems by combining the best translated chunks.

Original	Mr. <u>Pierce</u> said <u>Elcotel</u> should realize a minimum of \$10 of recurring net earnings for each machine each month
LogoMedia	El Sr Pierce dijo que Elcotel debe ganar <u>a minimum of</u> \$10 de ganancias netas se repitiendo para cada máquina todos los meses
Systran	Sr. <u>Elcotel</u> dicho <u>Pierce</u> debe realizar un mínimo de \$10 de las ganancias netas que se repiten para cada máquina cada mes.
SDL	Sr <u>Perfora</u> dijo que Elcotel debe darse cuenta de un mínimo de \$10 de ganancias netas periódicas para cada máquina cada mes
TB _{MEI}	El Sr Pierce dijo Elcotel debe realizar un mínimo de \$10 de las ganancias netas que se repiten para cada máquina cada mes.

Table 7.3: Result of TB_{MEI} vs. baseline MT engines on the example sentence ‘Mr. Pierce said Elcotel should realize a minimum of \$10 of recurring net earnings for each machine each month.’

In the translation of the example sentence in Table 7.3, LogoMedia leaves ‘a minimum of’ untranslated and uses a grammatically incorrect gerund ‘se repitiendo’. Systran switches the target positions of ‘Pierce’ and ‘Elcotel’, which severely distorts the accuracy of the translation. SDL interprets ‘Pierce’ as a verb ‘Perfora’, which makes the translation unintelligible. The MEMT combination, however, combines the best parts of each engine and is both accurate and relatively fluent.

7.4 Experimental Results and Analysis

7.4.1 Experimental Setup

To test the performance of TransBooster as an MEMT interface, we rely on the three standard automatic evaluation metrics (BLEU, NIST and GTM) described in Section 3.4.1 on page 27. The translated gold-standard test set against which the scores are calculated is the same 800-sentence test set as introduced in Section 3.4.2 and used in

Chapter 6

We experimented with three different syntactic analyses of the test set as input to our algorithm:

1. The original human parse-annotated Penn-II Treebank structures.
2. The output parses of the test set by (Charniak, 2000).
3. The output parses of the test set by (Bikel, 2002).

In each of these three cases, our algorithm decomposes the input into chunks and combines the chunk outputs of the MT engines as described in Section 7.3.2. As in the previous chapter, we are not merely interested in the absolute scores of the MEMT algorithm, but we also want to measure the impact on the results of the necessarily ‘imperfect’ parser output of (Charniak, 2000) and (Bikel, 2002) with respect to the ‘perfect’ human parse-annotated sentences of the Penn Treebank.

In addition to comparing the MEMT output to the three baseline MT systems, we also compute evaluation scores for the output of TransBooster interfaced with only one of baseline systems at each time (TB_{SEI}). This allows us to measure the impact of the effect on the scores of the multi-engine approach versus the possible individual score enhancements of TransBooster.

For practical reasons, contrary to the evaluations in Chapter 6, we have refrained from performing a detailed manual analysis of the output, given the many different system combinations and outputs involved.

7.4.2 Results

Table 7.4 contains the automatic evaluation scores for the three baseline MT systems against which we will compare the TB_{MEI} and TB_{SEI} scores in the following sections.

At the end of each of the following three sections (Section 7.4.2.1: ‘Human parse-annotated input’, Section 7.4.2.2: ‘Input parsed by (Charniak, 2000)’, and Section 7.4.2.3: ‘Input parsed by (Bikel, 2002)’) we will explain the relative contribution of the different chunk selection heuristics to the overall MEMT score. While performing the experiments, we noticed that comparable chunk translations with a different lexical contents *never*

	BLEU	NIST	GTM
LogoMedia	0.3140	7.3272	0.5627
Systran	0.3003	7.1674	0.5553
SDL	0.3039	7.2735	0.5657

Table 7.4: Results of the three baseline MT systems on the 800-sentence test set: absolute scores (cf. Table 6.3 in Chapter 6) on page 101).

received the same Language Model score. Therefore, in practice, the confidence score heuristic was never used. In order to verify the impact of this last heuristic on the test results, we decided to select the chunk with the best Language Model score only if the difference between the best and second best Language Model scores was smaller than a predefined threshold `p_LMDifference`. After experimenting with `p_LMDifference = 10, 5, 2, 1, and 0`, we found that the optimal results were produced for `p_LMDifference = 0`. Therefore, in each of the three following sections, only the Majority Voting and Language Model scores were used to select the optimal chunk.

7.4.2.1 Human parse-annotated input

Table 7.5 contains the absolute scores of TB_{MEI} and TB_{SEI} for the human parse-annotated version of the 800-sentence test set. Although we obtained the TB_{SEI} scores by applying exactly the same procedure as followed in Chapter 6, the TB_{SEI} results in this chapter slightly differ from the ones reported in the previous one. The reason for this difference is that, while the scores reported in Chapter 6 correspond to the latest optimal version of the algorithm, TB_{MEI} was implemented on a previous, intermediate version of the TransBooster algorithm. This slight difference in absolute scores is not an inconvenience, since the central research question of this chapter is to find out whether TransBooster has potential as an interface to MEMT. In other words, in this analysis, we are mainly interested in the relative scores of TB_{MEI} vs. TB_{SEI} and each of the baseline MT systems, which are reported in Table 7.6. TB_{MEI} improves relative to the baseline MT engines by between 5.9%-10.7% BLEU score, 5.2%-7.5% NIST score and 2.8%-4.8% GTM score. The relative improvements of TB_{MEI} with respect to TB_{SEI} are 5.3%-10.9% BLEU score, 5.0%-7.2% NIST score and 3.3%-4.8% GTM score.

The TB_{MEI} results can be explained by a combination of two different factors:

	BLEU	NIST	GTM
TB MEMT	0.3326	7.7119	0.5821
TB LogoMedia	0.3157	7.3383	0.5623
TB Systran	0.2998	7.1910	0.5553
TB SDL	0.3049	7.3169	0.5635

Table 7.5: TB_{MEI} vs TB_{SEI} : absolute scores for human parse-annotated input

	BLEU(%)	NIST (%)	GTM(%)
LogoMedia	105.9	105.2	103.4
TB LogoMedia	105.3	105.0	103.5
Systran	110.7	107.5	104.8
TB Systran	110.9	107.2	104.8
SDL	109.4	106.0	102.8
TB SDL	109.0	105.3	103.3

Table 7.6: TB_{MEI} vs. TB_{SEI} and baseline systems: relative scores for human parse-annotated input.

1. TB_{MEI} improves thanks to the benefits of a multi-engine approach to MT, in which the selection procedure (cf. Section 7.3.2) eliminates bad chunk translations. This is a characteristic shared by all MEMT approaches. In terms of a general MEMT architecture, the main novelty of our approach is that TB_{MEI} prepares the input sentence for multi-engine processing from the input side, unlike all other previous MEMT approaches.
2. TB_{MEI} improves thanks to the benefits of the recursive decomposition characteristics of TransBooster. In other words, the decomposition of the input sentence into syntactically simpler chunks allows the individual MT systems to improve on their *own* translations.

In order to obtain a more accurate idea of the relative contribution of each of these factors to the overall improvements, it is important to analyse the differences between TB_{MEI} and TB_{SEI} . Table 7.7 contains the relative results of TB_{SEI} vs. the three baseline MT systems

The fact that the relative results of TB_{SEI} in Table 7.7 are significantly lower than the relative results of TB_{MEI} in Table 7.6 seems to indicate that the most important contribution to the success of TB_{MEI} comes from the general benefits of a multi-engine

	BLEU(%)	NIST (%)	GTM(%)
LogoMedia	100.5	100.1	99.9
Systran	99.8	100.3	100
SDL	100.3	100.6	99.6

Table 7.7: TB_{SEI} vs. baseline systems relative scores for human parse-annotated input.

approach to MT, rather than the recursive decomposition characteristics of TransBooster. This observation does not, however, weaken the finding that TransBooster can be used as a valid MEMT interface, as is clearly shown by the results in Table 7.6. It merely indicates that it is mainly the chunking component of TransBooster, rather than its potential to help an MT system improve its own translations, which leads to the overall improvements.

The figures in Table 7.8 show the relative contribution of each of the different chunk selection heuristics to the overall MEMT score for the pre-parsed Penn-II input. On the entire 800-sentence test set, 5258 different chunk comparisons were performed. In 64.7% of the cases, the optimal chunk was selected using Majority Voting. In the remaining 35.3% of the comparisons, the best chunk was selected relying on the Language Model score. Since the optimal results were obtained with $p_{LMDifference} = 0$ (cf. explanation on page 145), the MT confidence score was never used.

Selection heuristic	Nr. comparisons	%
Majority Voting	3404	64.7
Language Model	1854	35.3
Confidence Score	0	0
Total	5258	100

Table 7.8: Relative contribution of each of the selection heuristics for the results in Table 7.5

7.4.2.2 Input parsed by (Charniak, 2000)

Table 7.9 contains the absolute scores of TB_{MEI} and TB_{SEI} for the output of (Charniak, 2000) on the 800-sentence test set. Table 7.10 contains the relative scores of TB_{MEI} vs. TB_{SEI} and each of the baseline MT systems, on the output of (Charniak, 2000) on the 800-sentence test set.

TB_{MEI} improves relative to the baseline MT engines between 2.7%-7.3% for BLEU, 3.8%-6.1% for NIST and 1.6%-3.6% for GTM. The relative improvements of TB_{MEI} with respect to TB_{SEI} are 3.7%-8.7% BLEU score, 4.4%-6.5% NIST score and 2.4%-4.1% GTM score.

	BLEU	NIST	GTM
TB MEMT	0.3225	7.6080	0.5753
TB LogoMedia	0.3108	7.2860	0.5604
TB Systran	0.2966	7.1393	0.5524
TB SDL	0.3004	7.2842	0.5615

Table 7.9: TB_{MEI} and TB_{SEI} . absolute scores for input parsed by (Charniak, 2000)

	BLEU(%)	NIST(%)	GTM(%)
LogoMedia	102.7	103.8	102.2
TB LogoMedia	103.7	104.4	102.6
Systran	107.3	106.1	103.6
TB Systran	108.7	106.5	104.1
SDL	106.1	104.5	101.6
TB SDL	107.3	104.4	102.4

Table 7.10: TB_{MEI} vs TB_{SEI} and baseline systems' relative scores for input parsed by (Charniak, 2000)

The figures in Table 7.11 show the relative contribution of each of the different chunk selection heuristics to the overall MEMT score for the pre-parsed Penn-II input. On the entire 800-sentence test set, 5223 different chunk comparisons were performed. In 65.1% of the cases, the optimal chunk was selected using Majority Voting. In the remaining 34.9% of the comparisons, the best chunk was selected relying on the Language Model score. Since the optimal results were obtained with $p_LMDifference = 0$ (cf. explanation on page 145), the MT confidence score was never used.

Selection heuristic	Nr. chunks	%
Majority Voting	3402	65.1
Language Model	1821	34.9
Confidence Score	0	0
Total	5223	100

Table 7.11: Relative contribution of each of the selection heuristics for the results in Table 7.9.

7.4.2.3 Input parsed by (Bikel, 2002)

Table 7.12 contains the absolute scores of TB_{MEI} and TB_{SEI} for the output of (Bikel, 2002) on the 800-sentence test set. Table 7.13 contains the relative scores of TB_{MEI} vs. TB_{SEI} and each of the baseline MT systems, on the output of (Bikel, 2002) on the 800-sentence test set.

TB_{MEI} improves relative to the baseline MT engines between 2.3%-7.0% for BLEU, 3.8%-6.1% for NIST and 1.7%-3.6% for GTM. The relative improvements of TB_{MEI} with respect to TB_{SEI} are 2.9%-8.8% BLEU score, 4.1%-6.3% NIST score and 2.5%-4.2% GTM score.

The figures in Table 7.14 show the relative contribution of each of the different chunk selection heuristics to the overall MEMT score for the pre-parsed Penn-II input. On the entire 800-sentence test set, 5178 different chunk comparisons were performed. In 63.7% of the cases, the optimal chunk was selected using Majority Voting. In the remaining 36.3% of the comparisons, the best chunk was selected relying on the Language Model score. Since the optimal results were obtained with $p_LMDifference = 0$ (cf. explanation on page 145), the MT confidence score was never used.

	BLEU	NIST	GTM
TB MEMT	0.3215	7.6079	0.5758
TB LogoMedia	0.3122	7.3032	0.5589
TB Systran	0.2953	7.1517	0.5521
TB SDL	0.3006	7.2891	0.5614

Table 7.12: TB_{MEI} and TB_{SEI} : absolute scores for input parsed by (Bikel, 2002)

	BLEU(%)	NIST(%)	GTM(%)
LogoMedia	102.3	103.8	102.3
TB LogoMedia	102.9	104.1	103.0
Systran	107.0	106.1	103.6
TB Systran	108.8	106.3	104.2
SDL	105.7	104.5	101.7
TB SDL	106.9	104.3	102.5

Table 7.13: TB_{MEI} vs TB_{SEI} and baseline systems: relative scores for input parsed by (Bikel, 2002)

Selection heuristic	Nr chunks	%
Majority Voting	3299	63.7
Language Model	1879	36.3
Confidence Score	0	0
Total	5178	100

Table 7.14: Relative contribution of each of the selection heuristics for the results in Table 7.12.

As expected, the scores based on parser-based output are slightly lower than the scores based on human parse-annotated sentences, with minimal differences between scores produced on output of (Charniak, 2000) and (Bikel, 2002). Even so, the overall scores of TB_{MEI} on parser output outperform both the baseline systems and TB_{SEI} with fairly large (statistically significant) margins, making TB_{MEI} an interesting alternative to previous developed MEMT approaches

7.5 Summary

In this chapter, we have explained how TransBooster, extended with a selection procedure based on majority voting, a language model score and a confidence score assigned to each baseline MT engine, can be used as a successful interface to Multi-Engine Machine Translation. The main novelties of our approach are the following: (i) the input sentence is prepared for multi-engine processing, in contrast to previous proposals in this area, which exclusively rely on target (sub-)sentence combination, (ii) TransBooster’s decomposition algorithm has the potential to help the individual baseline MT engines improve on their own individual contributions to the MEMT output. We reported statistically significant relative improvements of over 10% BLEU score in experiments (English→Spanish) carried out on an 800-sentence test set extracted from the Penn-II Treebank. We explained that the main factor underlying these improvements is the appropriateness to MEMT of TransBooster’s recursive chunking of the input.

Chapter 8

Conclusions

TransBooster is a novel approach designed to improve the translation quality of MT systems. TransBooster is not an MT engine itself: it acts on top of an already existing baseline MT system as a wrapper application. It simplifies complex input sentences by a recursive decomposition algorithm that transforms the original input into shorter chunks, which pose less challenges to the underlying MT system. This complexity reduction enables the baseline MT system to do what we think it does best, namely process a concise, syntactically simple skeleton with a reasonable expectation of a good translation. TransBooster guides the baseline system through the entire translation process by spoon-feeding it simple chunks and composing the output with the retrieved chunk translations.

In this thesis, we first introduced the rationale for recursive sentence decomposition in MT and compared the TransBooster approach to other MT paradigms. After reporting our initial experiments to determine the best form of Static Substitution Variables, we explained the developed TransBooster architecture in depth. We also reported on the development of a parallel, simpler TransBooster architecture (TB_{MarkII}) and explained the differences between the original TB_{MarkI} algorithm and TB_{MarkII} . We analysed the performance of TransBooster on three RBMT systems, one SMT system and one EBMT system using both automatic and manual evaluation measures. Finally, we investigated the possibility of using TransBooster as an MEMT interface.

The main findings of the research presented in this dissertation are the following:

- The TransBooster technology has the potential to improve on both rule-based and

data-driven MT systems

- The improvements induced by TransBooster are triggered by complexity reduction of the input.
- Most of the cases in which TransBooster deteriorates the original output are due to context distortion.
- The possible improvements depend on the baseline MT system used. The output produced by TransBooster shares many characteristics of the baseline output, but improves on lexical selection, homograph resolution, word order and agreement features.
- When evaluated on an 800-sentence test set randomly extracted from Section 23 of the Penn-II Treebank, TransBooster outperforms two of the three baseline RBMT systems (SDL and LogoMedia) and achieves similar results compared to the third system (Systran), both in terms of automatic evaluation as of manual evaluation results.
- The noise introduced by the use of state-of-the-art statistical parsers ((Charniak, 2000) and (Bikel, 2002)) has an expected negative impact on the improvements gained by complexity reduction. Despite a slight reduction in translation quality, the use of TransBooster on RBMT systems still leads to a modest increase in performance when (Charniak, 2000) is used as front-end parser.
- The improvements achieved by TransBooster on data-driven MT systems (both SMT and EBMT) seem to be more pronounced than the improvements on rule-based MT systems. There are two possible explanations for this: (i) data-driven MT systems benefit more from TransBooster's syntactic guidance than rule-based systems, and (ii) the baseline data-driven systems were possibly easier to improve on than the more performant rule-based systems used in the experiments.
- For the language pair used for evaluation purposes (English→Spanish), TB_{MarkI} achieves better results than TB_{MarkII} . This is due to (i) the larger scope of com-

plexity reduction of the TB_{MarkI} implementation, and (ii) the fact that the capacity of TB_{MarkII} to handle split pivots in target is not visible in Romance languages.

- TransBooster was successfully adapted as an MEMT interface, with reported relative improvements of up to 10% BLEU score over the baseline MT systems. These improvements are caused by the fact that TransBooster’s chunking algorithm effectively prepares the input sentence for multi-engine processing.

8.1 Future Work

There are a number of ways to extend the research presented in this dissertation:

The Static Substitution Variable (SSV) of a constituent is a simple string that, at best, shares certain syntactic characteristics with the substituted constituent. The outcome of the experiment in Section 4.3.4 showed that, even in a simplified environment, the syntactic and lexico-semantic differences between a range of SSVs and the original constituents can lead to distortions in the translation of the pivot and the placement of the satellites in target. Therefore, it is important to choose an SSV that is as similar as possible to the original. An avenue for further research could include optimising the SSVs used in this thesis (cf. Appendix D) by using information contained in ontologies combined with intelligent semantic similarity measures.

Another possibility to improve the output quality of TransBooster is the incorporation of named-entity recognition in the decomposition algorithm. In the current implementation, we use a simple heuristic based on the information provided by the Penn-II tags for proper nouns (*NNP* and *NNPS*) to decide when to keep an NP constituent from being translated, but we hypothesise that more sophisticated disambiguation methods will lead to further improvements in translation quality.

When using TransBooster as an MEMT interface, it would be interesting to see whether a word graph-based MEMT consensus at the level of the output chunks has the potential of improving our approach. Instead of simply selecting the best output chunk based on the described heuristics (cf. Section 7.3.2), an existing MEMT approach could be used to form a word-graph consensus translation at chunk level. Other avenues for further MEMT

research include replacing the similarity measure used in the selection procedure by an Edit Distance metric and experimenting with a variety of language models, similar to Nomoto (2004). In addition, one would expect an optimal MEMT system to contain baseline systems of different MT paradigms, so that the MEMT system can take advantage of the strengths of each individual approach. Accordingly, it would be interesting to experiment with TransBooster MEMT as a combination of RBMT, SMT and EBMT baseline systems.

Appendix A

Tags and Phrase Labels in the Penn-II Treebank

Tag Label	Tag Description
CC	Coordinating Conjunction
CD	Cardinal Number
DT	Determiner
EX	Existential there
FW	Foreign Word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal Pronoun
PRP\$	Possessive Pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, present participle

Continued on next page

Tag Label	Tag Description
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	WH-determiner
WP	WH-pronoun
WP\$	Possessive WH-pronoun
WRB	WH-adverb

Table A.1: Tag labels in the Penn-II Treebank.

Phrase Label	Phrase Description
ADJP	Adjectival Phrase
ADVP	Adverbial Phrase
CONJP	Conjunction Phrase
FRAG	Fragment
INTJ	Interjection
LST	List marker
NAC	Not a constituent
NP	Noun phrase
NX	N-bar (head of NP)
PP	Prepositional Phrase
PRN	Parenthetical
PRT	Particle
QP	Quantifier Phrase
RRC	Reduced relative clause
S	Declarative main clause
SBAR	Subordinate clause
SBARQ	Direct question
SINV	Inverted declarative sentence
SQ	Inverted yes/no question
UCP	Unlike Coordinated Phrase
VP	Verb Phrase
WHADJP	WH-adj phrase
WHADVP	WH-adv phrase
WHNP	WH-noun phrase
WHPP	WH-prep phrase
X	Unknown, uncertain or unbracketable

Table A 2: Phrase labels in the Penn-II Treebank.

Appendix B

Extended Pivot Selection per Category

CAT	Types	Basic Extended Pivot Treatment
ADJP	10	<p>pivot = (RB) + head + (IN/TO)</p> <p>Examples.</p> <p>ADJP = 'able to transfer money from the new funds' → pivot = 'able to'</p> <p>ADJP = 'still capable of serving on the bench' → pivot = 'still capable of'</p> <p>ADJP = 'big enough for one consultant to describe it as clunky' → pivot = 'big enough for'</p>
ADVP	1	<p>pivot = head + (IN)</p> <p>Examples:</p> <p>ADVP = 'up from Wednesday's Tokyo close of 143 08 yen' → pivot = 'up from'</p> <p>ADVP = 'down from 9 45% a week earlier' → pivot = 'down from'</p>
CONJP	0	<p>too small for decomposition.</p> <p>Examples:</p> <p>CONJP = 'as well as'</p>
FRAG	138	no clear pattern → default pivot selection.
INTJ	0	<p>too small for decomposition</p> <p>Examples.</p> <p>INTJ = 'so long'</p>
LST	1	too small for decomposition
NAC	2	<p>too small for decomposition</p> <p>Examples</p> <p>NAC = 'University of Vermont'</p>
NP	27	<p>Default treatment NP: if head of NP is non-terminal, pivot = (DT) + head + (IN). If head of NP is a terminal node, pivot = left-to-right concatenation of all children up to head.</p> <p>Examples</p> <p>NP = 'any research on smokers of the Kent cigarettes' → pivot = 'any research on'</p>

Continued on next page

CAT	Types	Basic Extended Pivot Treatment
		<p>NP = 'the risk factors that led to the company's decision' → pivot = 'the risk factors'</p> <p>NP = 'actual collections made until Dec. 31 of this year' → pivot = 'actual collections'</p> <p>NP = 'no useful information on whether users are at risk' → pivot = 'no useful information on'</p> <p>NP = 'the sale of four of its TV stations for \$ 120 million' → pivot = 'the sale of'</p> <p>NP = 'the types of watches that now will be eligible for duty-free treatment' → pivot = 'the types of watches'</p> <p>NP = 'the right to increase its interest to 70%' → pivot = 'the right to'</p>
NX	6	<p>pivot = head + (IN)</p> <p>NP = 'host of a popular television talk show' → pivot = 'host of'</p>
PP	1	if head is not already attached to mother node pivot → default pivot selection
PRN	19	parenthetical → treat head
PRN		<p>Examples</p> <p>'- the symmetry of geometrical figures, metric measurement of volume, or pie and bar graphs, for example -' → pivot = 'the symmetry of'</p>
PRT	0	too small for decomposition
QP	2	<p>too small for decomposition</p> <p>Examples</p> <p>QP = 'more than three times'</p>
RRC	7	<p>too small for decomposition</p> <p>Examples.</p> <p>RRC = 'currently on the market'</p>
S	12	<p>→ VP</p> <p>Examples.</p> <p>S = 'At the end of the day, 251.2 million shares were traded' → pivot = 'were traded'</p> <p>S = 'The Dow fell 22 6% on Black Monday' → pivot = 'fell'</p> <p>S = 'This role reversal holds true, as well, for his three liberal and moderate allies' → pivot = 'holds true'</p> <p>S = 'Certainly, the recent drop in prices doesn't mean Manhattan comes cheap' → pivot = 'doesn't mean'</p> <p>S = 'The four are prepared to accept this new role' → pivot = 'are prepared to'</p> <p>S = 'waves of selling continued to hit stocks themselves on the Big Board' → pivot = 'continued to hit'</p> <p>S = 'Justice Blackmun, who will turn 81 next month, also seems feisty about his new role' → pivot = 'seems feisty about'</p>
SBAR	2	Sentential complement clauses are treated by attaching the complementizer to the verbal pivot and continuing with the composition of S. SBARs modifying a nominal antecedent are not decomposed

Continued on next page

CAT	Types	Basic Extended Pivot Treatment
		Examples S = 'A P&G spokeswoman confirmed that shipments to Phoenix started late last month' → pivot = 'confirmed that' S = 'Indeed, a random check Friday didn't seem to indicate that the strike was having much of an effect on other airline operations' → pivot = 'didn't seem to indicate that'
SBARQ	30	Do not decompose: limited amount of occurrences (241 sentences in sections 01–22 of Penn Treebank)
SINV	21	→ S Examples SINV = ' "We braced for a panic," said one top floor trader ' → pivot = 'said' SINV = 'Hardest hit are what he calls "secondary" sites that primarily serve neighborhood residents ' → pivot = 'Hardest hit are'
SQ	77	Do not decompose: pivot difficult to extract due to inversion and limited amount of occurrences (405 sentences in sections 01–22 of Penn Treebank)
UCP	106	coordination → pivot = CC Examples UCP = 'the largest maker of personal computer software and generally considered an industry bellwether' → pivot = 'and'
VP	72	Recursive pivot determination. Basics: string together verbal lexical categories ('MD', 'VBD', 'VBP', 'VBZ', 'VBN', 'VBG', 'VB'), including certain intermediate nodes (e.g. ADVP, ADJ-PRD, RP). If VBN or VBG preceded by 1 other node, include this node, regardless of length. Attach 'TO' where necessary. (cf examples of sentential categories)
WHADJP	0	too small for decomposition WHADJP = 'how many'
WHADVP	0	too small for decomposition WHADVP = 'when'
WHNP	0	too small for decomposition WHNP = 'which'
WHPP	0	too small for decomposition WHPP = 'of which'
X	20	Do not decompose: no clear pattern

Table B.1: Nr. of rule types (covering 85% of rule tokens) and basic extended pivot treatment for non-terminal nodes in the Penn-II Treebank
Parentheses indicate optional categories

Appendix C

ARG/ADJ distinction heuristics

Remarks concerning the information contained in Tables C.1 and C.2:

- The ARG/ADJ distinction heuristics are based on (Hockenmaier, 2003) and a manual inspection of the most frequent rule-types accounting for 85% of rule token expansions per non-terminal in the Penn Treebank, as is explained in Section 5.2.3.
- Nodes that have been assigned ‘head’ during the previous head-finding procedure are not taken into account for ARG/ADJ assignment.
- For each node N, all children are scanned from left to right. For each child C, the following three different strategies are considered:
 1. If C conforms to the description in Table C.1, Section A, then assign the corresponding ARG/ADJ distinction and move on to the next child. If not, go to step 2.
 2. If C conforms to the description in Table C.1, Section B, then assign the corresponding ARG/ADJ distinction and move on to the next child. If not, go to step 3.
 3. If C conforms to the description in Table C.2, then assign the corresponding ARG/ADJ distinction and move on to the next child. If not, assign the default p_SatDefault and move on to the next child. Note that in Table C.2, the column entitled ‘mother’ refers to node N, and the column entitled ‘CAT’ refers to the child node C.
- $X \rightarrow A B$ X expands into A and B
 $X < A$ X dominates A
 $X \rightarrow (A < B) C$ X expands into A and C. A dominates B.

CAT	TAG	ARG/ADJ	Comments
Section A			
CD		adj	unless when preceded by \$, in which case CD is arg, as in (QP (\$ \$) (CD 16) (CD million)).

Continued on next page

CAT	TAG	ARG/ADJ	Comments
CONJP		arg	Note that CONJPs in the Penn Treebank tend to dominate a limited amount of lexical items, as in 'rather than' or 'as well as'. unless when preceded by \$, in which case QP is arg, as in (NP (\$ \$) (QP (26 CD) (million CD)) (-NONE- *U*))
PRN		adj	
PRT		arg	
QP		adj	
RRC		adj	
SINV		arg	
WHADJP		arg	
WHADVP		arg	
WHNP		arg	
WHPP		arg	
X		arg	
Section B			
	ADV	adj	PP-DIR and ADVP-DIR under VP are classified as arg. PP-TMP under ADJP are classified as arg.
	BNF	adj	
	CLR	arg	
	DIR	adj	
	LOC	adj	
	MNR	adj	
	NAC	adj	
	PRD	arg	
	PRP	adj	
	TMP	adj	
	TPC	arg	

Table C.1: ARG/ADJ distinction heuristics per category, independent of the mother node.

Mother	CAT	ARG/ADJ	Comments
ADJP	NP	arg	if ADJP→ADJP PP if ADJP→VBN PP default SBAR = adj if introduced by 'than', 'as' or 'so'.
	PP	arg	
	PP	arg	
	PP	adj	
	S	arg	
	SBAR	arg	
	default	p_SatDefault	
ADVP	NP	arg	if left of head if head = 'than' default
	PP	adj	
	PP	adj	
	PP	arg	

Continued on next page

Mother	CAT	ARG/ADJ	Comments
	SBAR SBAR SBAR default	adj adj arg p_SatDefault	if preceded by comma if head = 'than', 'as', 'so', 'before' or 'which'. default
CONJP			not relevant since node is translated in entirety.
FRAG			not relevant since node is translated in entirety.
NP	JJ ADJP NNP NNPS NP PP PP PP PP S S SBAR default	adj adj arg arg arg arg adj arg adj arg adj adj p_SatDefault	except a list of 'determiner-like' JJs as 'many', 'much', 'more', ... if head NP = NNP or NNPS, otherwise adj. if head NP = NNP or NNPS, otherwise adj. unless apposition, in which case adj. NP→NP PP(arg) PP(adj) NP→NP , PP for a number of lexical cases such as 'a lot of', 'a kind of', 'a type of', ... default. NP→DT NN S default.
NX	PP default	adj p_SatDefault	
NAC			not relevant since node is translated in entirety
PP	ADVP ADJP NP S PP default	adj arg arg arg arg p_SatDefault	
PRN			not relevant since node is translated in entirety
S	ADVP NP PP RB RB S SBAR default	adj arg adj arg adj arg adj p_SatDefault	if negation. default
SQ	VP default	arg p_SatDefault	
SBAR	ADVP NN	adj arg	

Continued on next page

Mother	CAT	ARG/ADJ	Comments
	S SBARQ SINV SQ VP RB RB default	arg arg arg arg arg arg adj p_SatDefault	if negation. otherwise.
VP	ADJP NP NP PP PP PP S S SQ SBAR SBAR SBARQ XP default	arg adj arg adj arg p_SatDefault adj arg arg adj arg arg arg p_SatDefault	if apposition. default if PP-EXT. if first node = VBN default. VP→S , S (adj). default if preceded by comma and first child = WHNP, ADVP, RB or IN ('on' or 'with'). default.
WHNP			not relevant since node is translated in entirety.
WHADJP			not relevant since node is translated in entirety.
WHADVP			not relevant since node is translated in entirety.
WHPP			not relevant since node is translated in entirety.
X			not relevant since node is translated in entirety.
Default		p_SatDefault	

Table C.2: ARG/ADJ distinction heuristics per category, dependent of the mother node

Appendix D

Static Substitution Variables per Category

Remarks concerning the information contained in Table D.1:

- The table contains an exhaustive overview of how SSVs are generated for *all* possible satellites, even if certain types of satellite replacements do not (often) occur in practice due to pivot extensions. For example, despite the fact that a preposition is often attached to the preceding verb during the formation of the verbal pivot, a general treatment for PP substitution has been implemented. Substitutions like these are triggered in case an error occurs in the pivot extension procedure and have been included for reasons of completeness. Extremely rare cases are marked with a footnote.
- Examples mark the SSV substitution of the satellite category instance (displayed inside $[]_{SSV}$). Certain examples contain lexical items outside the syntactic environment treated for reasons of clarity.
- For each SSV displayed in this table, three syntactically similar but lexically different strings are available (cf. Section 5.2.4.1). These alternative strings are not included in the table so as not to clutter the general overview.
- $X \rightarrow A B$ X expands into A and B
 $X < A$ X dominates A
 $X \rightarrow (A < B) C$ X expands into A and C. A dominates B.

CAT	TAG	Environment	SSV
ADJP	-	NP \rightarrow NP ADJP ‘Issues [central to the increasingly tense trade debate]’ \rightarrow ‘Issues [similar to the house] _{SSV} ’	‘similar to the house’
	-	default ‘[green, black, red and white] stripes’ \rightarrow ‘[red] _{SSV} stripes’	‘red’
ADVP	-	default ‘The slowdown is taking hold [a lot more quickly and devastatingly than anyone had expected]’ \rightarrow ‘The slowdown is taking hold [quickly] _{SSV} ’	‘quickly’

Continued on next page

CAT	TAG	Environment	SSV
NP	EXT	head contains %	'10%'
		'surged [4 26, or about 0 94%]' → 'surged [10 %] _{SSV} '	
	EXT	head = NN	'a lot'
		'rose [a surprisingly moderate 0.2%]' → 'rose [a lot] _{SSV} '	
	EXT	head = NNS	'10 metres'
		'drop [an additional 2 5 feet]' → 'drop [10 metres] _{SSV} '	
	-	PP-LOC < NP (head = 'Chicago'	
		NNP/NNPS)	
		'in [Arizona, California, Louisiana and Maryland]' → 'in [Chicago] _{SSV} '	
	-	PP-LOC < NP	'the house'
		'in [an expensive high rise building]' → 'in [the house] _{SSV} '	
	-	PP-TMP < NP (head = '10 minutes'	
		NNS)	
		'during [the first nine months of the year]' → 'during [10 minutes] _{SSV} '	
		PP-TMP < NP	'tomorrow'
		'until [March, April or even May]' → 'until [tomorrow] _{SSV} '	
	-	head = PRP	mimic PRP ^a
		'[He]' → '[He] _{SSV} '	
	-	head = NN, det. article	'the boy'
		'[The young, short-term American employee]' → '[The boy] _{SSV} '	
-	head = NN, indet. article	'a cat'	
	'[A major U S producer and seller]' → '[A cat] _{SSV} '		
-	head = NN, mass noun	'sugar'	
	'[Some MCI. Communications Corp stock]' → '[Sugar] _{SSV} '		
-	head = NN (default)	'the boy'	
	'[Even the official Indianapolis 500 announcer]' → '[The boy] _{SSV} '		
-	head = NNS	'the swimmers'	
	'[The other two outside bidders]' → '[The swimmers] _{SSV} '		
-	head = NNP	'John'	
	'[The French film maker Claude Chabrol]' → '[John] _{SSV} '		
-	head = NNPS	'John and Alex'	
	'[Peter D. Hart Research Associates]' → '[John and Alex] _{SSV} '		
-	head = JJS	'most'	
	'[Most soybean and soybean-meal contracts]' → '[Most] _{SSV} '		
-	head = DT	mimic DT ^b	
	'[That]' → '[That] _{SSV} '		
PP	DTV	head = IN	'to the man'
		'an approach to offer [not only to Californians, but to all Americans]' → 'an approach to offer [to the man] _{SSV} '	
	DIR	head = IN ('to')	'to London'
	'fled [to Canada or some other sanctuary]' → 'fled [to London] _{SSV} '		
DIR	head = IN ('from')	'from London'	

Continued on next page

^aPersonal pronouns are left intact in cases in which they are not included in the pivot

^bDeterminers are left intact in cases in which they are not included in the pivot

CAT	TAG	Environment	SSV
	LOC	'Oil production [from Australia's Bass Straight Fields]' → 'Oil production [from London] _{SSV} ' head = IN	'in the house'
	MNR	'[in the rapidly growing field of bio-analytical instrumentation]' → '[in the house] _{SSV} ' head = IN	'with an apple'
	TMP	'[with large and expensive page bonuses]' → '[with an apple] _{SSV} ' head = IN	'after the meeting'
	-	'[after a loss to the Kansas City Chiefs yesterday]' → '[after the meeting] _{SSV} ' head = IN	mimic preposition ^a
	-	'[before entering restaurants, department stores and sports centres]' → '[before the holiday] _{SSV} ' head = VBG ('including')	'including the dog'
	-	'[including perhaps someone of your own staff]' → '[including the dog] _{SSV} ' head = VBG ('according')	'according to the woman'
	-	'[according to government figures released Wednesday]' → '[according to the woman] _{SSV} ' head = VBG ('following')	'following the meeting'
	-	'following the June 4 massacre in Beijing, which caused a sharp drop in Hong Kong prices' → '[following the meeting] _{SSV} ' head = VBG ('excluding')	'excluding the dog'
	-	'[excluding the hard-hit city of Los Gatos]' → '[excluding the dog] _{SSV} ' head = VBG ('depending')	'depending on the meeting'
	-	'[depending on the composition of the management team and the nature of its strategic plans]' → '[depending on the meeting] _{SSV} ' head = TO	'to the dog'
	-	'[to the troubled company's equity holders]' → '[to the dog] _{SSV} ' default	'in the house'
PRN	-	default	replace head PRN by appropriate SSV
		'Italian chemical giant Montedison S p.A. [,through its Montedison Acquisition N V. indirect unit,] began ...' → 'Italian chemical giant Montedison S p A [,through the man,] _{SSV} began . '	
UCP	-	default	replace UCP by SSV of first node
		'to be [in violation of Article II, and thus void and severable]' → 'to be [in the house] _{SSV} '	
S	TPC	-	'The man is sleeping'
	NOM	'[The total of 18 deaths from malignant mesothelioma, lung cancer and asbestosis was far higher than expected], the researchers said.' → '[The man is sleeping] _{SSV} , the researchers said'	'sleeping'
	ADV	'before anyone heard of [asbestos having any questionable properties]' → 'before anyone heard of [sleeping] _{SSV} ' head = VBG	'working in the garden'
		'standing around [deciding who would fly in what balloon and in what order]' → 'standing around [working in the garden] _{SSV} '	

Continued on next page

^aThe prepositions for PPs with more than 100 token occurrences in sections 01–22 of the Penn Treebank are mimicked.

CAT	TAG	Environment	SSV
	ADV	head = VBN	'founded in 1900'
		'[Filmed in lovely black and white by Bill Dill], the New York streets'	
		→ '[Founded in 1900] _{SSV} , the New York streets'	
	PRP	-	'to sleep'
		'resigned last year [to seek, unsuccessfully, a seat in Canada's parliament]' → 'resigned last year [to sleep] _{SSV} '	
	CLR	head = VBG	'working in the garden'
		'launched a suit, [seeking the withdrawal of Dunkin's poison pill rights and employee stock ownership plans]' → 'launched a suit, [working in the garden] _{SSV} '	
	CLR	-	'to sleep'
		'paid [to stand up at a Japanese plate]' → 'paid [to sleep] _{SSV} '	
	PRD	-	'a man'
		'The result has been [to seriously impair the rights of others unconnected with their dispute]' → 'The result has been [a man] _{SSV} '	
	HLN	-	'The man is sleeping'
		'Applause for "Sometimes Talk is the Best Medicine"' → 'Applause for "[The man is sleeping] _{SSV} "'	
	CLF	-	'It is the man'
		'[It is these 645,000 tons that are in question for this crop year], explained Judith Ganes' → '[It is the man] _{SSV} , explained Judith Ganes'	
	TTL	-	'The man is sleeping'
		'In reference to your Oct. 9 page-one article, "[Barbara Bush earns even higher ratings than the president,]" it is ...' → 'In reference to your Oct 9 page-one article, "[The man is sleeping,] _{SSV} " it is ...'	
	MNR	-	'working in the garden'
		'He earns his living [playing the double bass in classical music ensembles]' → 'He earns his living [working in the garden] _{SSV} '	
	SBJ	-	'to sleep'
		'[To suggest that a 10% drop in ozone by the middle of the next century would be negligible] is irresponsible and shortsighted.' → '[To sleep] _{SSV} is irresponsible and shortsighted'	
	TMP	-	'starting in 1990'
		'[Beginning in the first year of medical school], students learn' → '[starting in 1990] _{SSV} , students learn	
	-	S → NP-SBJ TO	'the boy to sleep'
		'causing [the index to decline for three consecutive months]' → 'causing [the boy to sleep] _{SSV} '	
	-	head = TO	'to sleep'
		'Longer maturities are thought [to indicate declining interest rates because they permit portfolio managers to maintain relatively higher rates]' → 'Longer maturities are thought [to sleep] _{SSV} '	
	-	S → NP-SBJ VBG	'the boy working in the garden'
		'I've had [a lot of people trying to sell me services to find out how big it is]' → 'I've had [the boy working in the garden] _{SSV} '	
	-	head = VBG	'working in the garden'
		'The stock, [having lost nearly a quarter of its value since Sept. 1], closed at \$34.375 share' → 'The stock, [working in the garden] _{SSV} closed at \$34.375 share'	

Continued on next page

CAT	TAG	Environment	SSV
	-	head = VBN '[Managed properly, and with a long-term outlook,] these can become ' → '[founded in 1900] _{SSV} , these can become . . '	'founded in 1900'
	-	head = VBD 'to indicate that [the strike was having much of an effect on other airline operations]' → 'to indicate that [the man was sleeping] _{SSV} '	'the man was sleeping'
	-	head = JJ 'it left [the market vulnerable to continued selling this morning]' → 'it left [the boy happy] _{SSV} '	'the boy happy'
	-	head = NN 'like [things just the way they are]' → 'like [things this way] _{SSV} '	'things this way'
	-	default	'the man is sleeping'
SINV	-	default	' "The dog is barking", said the man.'
SBAR	TMP	-	'after a week' '[When the dollar is in a free-fall], even central banks can't stop it' → '[after a week] _{SSV} , even central banks can't stop it'
	PRP	VP < SBAR	'because the man is/was sleeping' ^a 'perhaps [because I have a ferocious list of statutes to implement]' → 'perhaps [because the man is sleeping] _{SSV} '
	ADV	VP < SBAR	'if the man is/was sleeping' ^a 'We would be upset [if those kinds of services evolved into more general- interest, long-format programming]' → 'We would be upset [if the man is sleeping] _{SSV} '
	ADV	S < SBAR	'fortunately' '[Although the legality of these sales is still an open question], the disclosure couldn't be better timed' → '[fortunately] _{SSV} , the disclosure couldn't be better timed'
	LOC	VP < SBAR	'in the house' 'control pollution [where enterprises are state-owned and penalties are paid by the government]' → 'control pollution [in the house] _{SSV} '
	NOM	S < SBAR	'the boy' '[What becomes custom in the Bush administration] will become . . ' → '[the boy] _{SSV} will become . . '
	NOM	SBAR < what	'what the man found' 'Typical is [what happened to the price of ethylene, a major commodity chemical produced in vast amounts by many oil companies]' → 'Typical is [what the man found] _{SSV} '
	NOM	default	'that the man is sleeping' 'clearly show [why Cray research favoured the spinoff]' → 'clearly show [that the man is sleeping] _{SSV} '
	-	VP < (SBAR < (IN that))	'that the man is/was sleeping' ^a 'It seems to me [that a story like this breaks just before every important Cocom meeting]' → 'It seems to me [that the man is/was sleeping] _{SSV} '
	-	VP < (SBAR < (IN whether))	'whether the man is/was sleeping' ^a 'No one knows [whether the new posted prices will stick once producers and customers start to haggle]' → 'No one knows [whether the man is sleeping] _{SSV} '

Continued on next page

^aTense depends on tense of main clause.

CAT	TAG	Environment	SSV
	-	VP < (SBAR < (IN than))	'than the man' 'put a greater emphasis on quality [than they do in the US]' → 'put a greater emphasis on quality [than the man] _{SSV} '
	-	VP < (SBAR < (IN as))	'as the man' 'oppose funding [as does president Bush]' → 'oppose funding [as the man] _{SSV} '
	-	VP < (SBAR < (IN what))	'what the man found' 'The commissioner knows [what will happen down the road, in three to six months]' → 'The commissioner knows [what the man found] _{SSV} '
	-	NP < (SBAR < (TO to))	'to sleep' 'legislation [to protect foreign movie producers]' → 'legislation [to sleep] _{SSV} '
	-	NP < (SBAR < (IN where))	'where the man is sleeping' 'the office [where employees are assigned lunch partners]' → 'the office [where the man is sleeping] _{SSV} '
	-	VP < SBAR (default)	'that is/are sleeping' 'the brokerage company [that once did business as Merrill Lynch Commercial Real Estate]' → 'the brokerage company [that is sleeping] _{SSV} '
	-	ADJP ← JJ (SBAR < WHNP)	'how the man is/was sleeping' 'not sure [how many weapons they have in their arsenals]' → 'not sure [how the man is sleeping] _{SSV} '
	-	ADJP < SBAR (default)	'that the man is/was sleeping' 'stunned [that despite the bald-faced nature of her actions, she became something of a local martyr]' → 'stunned [that the man was sleeping] _{SSV} '
	-	PP → IN SBAR	'whether the man is/was sleeping' 'divided over [whether the United Nations Population Fund will receive any portion of these appropriations]' → 'divided over [whether the man is/was sleeping] _{SSV} '
	-	default	'that the man is sleeping'
VP ^a	-	NP < VP	'made in China' 'an exotic playground, [peopled mainly by Jewish eccentrics and the occasional Catholic]' → 'an exotic playground, [made in China] _{SSV} '
	-	VP < VP (head = TO)	'sleep' 'eager to [bring attention to the problem]' → 'eager to [sleep] _{SSV} '
	-	VP < VP (head = VB)	'sleep' 'the authority to [seize U.S. fugitives overseas without the permission of foreign governments]' → 'the authority to [sleep] _{SSV} '
	-	VP < VP (head = VBG)	'sleeping' 'the company had been [steadily lowering its accident rate and picking up trade-group safety awards]' → 'the company had been [sleeping] _{SSV} '
	-	VP < VP (head = VBN)	'paid' 'the effect has been [set up and shot down by different professors]' → 'the effect has been [paid] _{SSV} '
	-	VP < VP (head = VBZ)	'is sleeping' 'The company [is operating under Chapter 11 of the federal Bankruptcy Code]' → 'The company [is sleeping] _{SSV} '

Continued on next page

^a'VP < VP' SSV replacements rarely occur in practice due to verbal pivot extensions.

CAT	TAG	Environment	SSV
	-	VP < VP (head = VBP) " → 'eat an apple'	'eat an apple'
	-	VP < VP (head = VBD) 'The president has not [said before that the country wants half the debt forgiven]' → 'The president has not [slept] _{SSV} '	'slept'
WHADJP	-	WHNP < WHADJP (SG) '[how much credibility and experience]' → '[how much] _{SSV} '	'how much'
	-	WHNP < WHADJP (PL) '[how many company mail rooms]' → '[how many] _{SSV} '	'how many'
	-	default	'how much'
WHADVP	-	default '[precisely when and where]' → '[when] _{SSV} '	'when'
WHPP	-	default No occurrences	'in which'
default	-	default 'The man, a long-time rival of Bill Gates, likes fast and confidential deals' → '[SAT1] _{SSV} likes [SAT2] _{SSV} '	'SAT1'-'SAT9' ^a

^aBackoff to non-word strings if SSV is not selected in a particular syntactic environment or if all alternatives for the same category-environment pair have been used.

Table D.1 Static Substitution Variables per Category.

Appendix E

Static Context Templates per Category

Remarks concerning the information contained in Table E.1:

- For a specific satellite category, all occurrences in a syntactic environment that is not contained in the table do not require embedding in a static context template, unless otherwise specified by the word ‘default’ in the column *Environment*. In the latter case, all occurrences in a syntactic environment that is not contained in the table conform to the context specified in the row containing ‘default’.
- The table contains an exhaustive overview of how static context templates are generated for *all* possible satellites, even if certain types of satellites do not (often) occur in practice due to pivot extensions. For example, despite the fact that a preposition is often attached to the preceding verb during the formation of the verbal pivot, a general treatment for PP embedding has been implemented. Template insertions like these are triggered in case an error occurs in the pivot extension procedure and have been included for reasons of completeness. Extremely rare cases are marked with a footnote.
- Examples contain the satellite to be substituted (displayed inside []_{SAT}), the original context and the new context (displayed inside []_C).
- $X \rightarrow A B$ X expands into A and B
 $X < A$ X dominates A
 $X \rightarrow (A < B) C$ X expands into A and C. A dominates B.

CAT	TAG	Environment	Context
ADJP	-	NP < ADJP	NP _{rep} ^a
	-	default	‘[it seems] _C ADJP’
			‘ “progressive education” as it was once called is [far more interesting and agreeable to teachers than is disciplined instruction] _{ADJP} ’ → ‘[it seems] _C [far more interesting and agreeable to teachers than is disciplined instruction] _{ADJP} ’

Continued on next page

^aFor NP_{rep} consult end of Table

CAT	TAG	Environment	Context
ADVP	TMP	all	'[the man sings a song] _C ADVP'
	MNR	all	'[the man does it] _C ADVP'
	LOC	all	'Mr Bush has been criticized regularly at home for moving [too slow and cautiously] _{ADVP} ' → '[the man does it] _C [too slow and cautiously] _{ADVP} '
		all	'[the man lives] _C ADVP' the tremor was centered near Hollister, southeast of San Francisco, and was felt [as far as 200 miles away] _{ADVP} ' → '[the man lives] _C [as far as 200 miles away] _{ADVP} '
-	default	'[the man is sleeping] _C ADVP' 'one auto-industry union leader said that they tried to build it [somewhere else in Europe besides the U K] _{ADVP} ' → '[the man is sleeping] _C [somewhere else in Europe besides the U K] _{ADVP} '	
CONJP	-	default	CONJP [a man] _C '[as well as] _{CONJP} regional matters such as transportation and telecommunications → [as well as] _{CONJP} [a man] _C
NP	SBJ	S < NP (head = PRP)	PRPs are included in pivot ^a
	SBJ	S < NP (head = NN or NNP)	'NP [is sleeping] _C ' '[Pierre Vincken, 61 years old] _{NP} will join the board as a nonexecutive director Nov 29' → '[Pierre Vincken, 61 years old] _{NP} [is sleeping] _C '
	SBJ	S < NP (head = NNS or NNPS)	'NP [are sleeping] _C ' '[four of the five surviving workers] _{NP} have asbestos-related diseases, including three with recently diagnosed cancer' → '[four of the five surviving workers] _{NP} [are sleeping] _C '
	SBJ	S < NP (default)	'NP [is sleeping] _C '
	MNR	VP < NP	'[the man slept] _C NP' 'the thought of a living player selling his checks rubs some people [the wrong way] _{NP} ' → '[the man slept] _C [the wrong way] _{NP} '
	MNR	S < NP	'[, the man slept] _C NP' '[that way] _{NP} investors can essentially buy the funds without paying the premium' → '[that way] _{NP} [, the man slept] _C '
	TMP	VP < NP	'[the man slept] _C NP' 'the monthly sales have been setting records [every month since March] _{NP} ' → '[the man slept] _C [every month since March] _{NP} '
	TMP	S < NP	'[, the man slept] _C NP' '[late yesterday] _{NP} Georgia Gulf said it reviewed the proposal as well as interests from third parties' → '[late yesterday] _{NP} [, the man slept] _C '
	-	PP → VBG (PP < NP)	'[according to] _C NP' 'these materials are nothing short of sophisticated crib sheets, according to [some recent academic research] _{NP} ' → '[according to] _C [some recent academic research] _{NP} '
	-	PP → VBN (PP < NP)	'[compared with] _C NP' 'Sterling's firm tone, combined with [a steady opening on Wall Street] _{NP} also tempted some investors ..' → '[compared with] _C [a steady opening on Wall Street] _{NP} '
	-	PP → VBG NP	'[including] _C NP' 'Jaguar shares skyrocketed yesterday, following [their temporary suspension on London's Stock Exchange] _{NP} ' → '[including] _C [their temporary suspension on London's Stock Exchange] _{NP} '

Continued on next page

^aA context for PRPs is meaningless due to the fact that in the vast majority of cases, PRPs are not explicitly expressed in Spanish (zero-subject language).

CAT	TAG	Environment	Context
	-	PP → JJ IN NP 'sales rose 5% amid good growth in selected areas such as [banks and trading companies] _{NP} ' → '[such as] _C [banks and trading companies] _{NP} '	'[such as] _C NP'
	-	PP → IN IN NP 'that includes all the gas consumed in Ontario and Quebec, along with [the bulk of Canadian gas exports] _{NP} ' → '[because of] _C [the bulk of Canadian gas exports] _{NP} '	'[because of] _C NP'
	-	PP < NP (default) 'the strike was having much of an effect on [other airline operations] _{NP} ' → '[the man dances with] _C [other airline operations] _{NP} '	'[the man dances with] _C NP'
	-	NP → NP , NP (,) ' according to Brooke T. Mossman, [a professor in pathology] _{NP} ' → '[the man,] _C [a professor in pathology] _{NP} '	'[the man,] _C NP'
	-	VP < NP if NP = DOBJ: 'the man is (not) eating] _C ' if NP = predicative: zero context	
		'last month, the company's stock funds have averaged [a staggering gain of 25%] _{NP} ' → '[the man is eating] _C [a staggering gain of 25%] _{NP} ' 'after the voting debacle in parliament, I certainly wouldn't expect [an immediate resolution to anything] _{NP} ' → '[the man is not eating] _C [an immediate resolution to anything] _{NP} ' '[Mr. Vinken is] [chairman of Elsevier N V] _{NP} ' → '[] _C [chairman of Elsevier N V] _{NP} '	
PP	TMP	all 'compound yields assume reinvestment of dividends and that the current yield continues [for a year] _{PP} ' → '[the man sings a song] _C [for a year] _{PP} '	'[the man sings a song] _C PP'
	-	NP → NP , PP 'but a takeover battle opens up the possibility of a bidding war, [with all that implies] _{PP} ' → '[the man,] _C [with all that implies] _{PP} '	'[the man,] _C PP'
	-	NP < PP NP _{rep} ^a	
	-	VP < PP 'Pierre Vinken, 61 years old, will join the board [as a nonexecutive director] _{PP} ' → '[the man is sleeping] _C [as a nonexecutive director] _{PP} '	'[the man is sleeping] _C PP'
	-	S < PP '[in the new position] _{PP} he will oversee Mazda's US sales, service, parts and marketing operations' → '[in the new position] _{PP} [the man is sleeping] _C '	'PP [the man is sleeping] _C '
PRN	-	NP < PRN S < NP PRN	context = NP _{rep} ^a context = NP _{rep} ^a
QP	-	NP < QP '[no fewer than 24] _{QP} country funds have been launched or registered with regulators' → '[no fewer than 24] _{QP} [men] _C '	'QP [men] _C '
RRC	-	NP (head plural) < RRC 'together with the 3.6 million shares [controlled by management directors] _{RRC} ...' → '[the men,] _C [controlled by management directors] _{RRC} '	'[the men,] _C RRC'
	-	NP (head singular) < RRC ' "He makes snap judgements," says Kiyotaka Kori, [the art gallery's manager and Mr Morishita's secretary] _{RRC} ' → '[the man,] _C [the art gallery's manager and Mr Morishita's secretary] _{RRC} '	'[the man,] _C RRC'

Continued on next page

^aFor NP_{rep} consult end of Table

CAT	TAG	Environment	Context
S	NOM	all	'S [is good] _C ' 'a Commonwealth Edison spokesman said that [tracking down the two million customers] _S would be an administrative nightmare' → '[tracking down the two million customers] _S [is good] _C '
	ADV	S before comma	'S [, the man is sleeping] _C ' '[standing on a shaded hill] _S , the school has educated many of South Carolina's best and brightest' → '[standing on a shaded hill] _S [, the man is sleeping] _C '
	ADV	S after comma	'[the man is sleeping,] _C S' 'prior to his term, a teacher bled to death in the halls, [stabbed by a student] _S ' → '[the man is sleeping,] _C [stabbed by a student] _S '
	-	PP → IN S	Context = original preposition (IN) 'spending on private construction was off 2.6%, with [no sector showing strength] _S ' → '[with] _C [no sector showing strength] _S '
SBAR	ADV	VP < SBAR	'[the man is/was sleeping] _C SBAR' 'moreover, there have been no orders for the Cray-3 so far [though the company is talking with several prospects] _{SBAR} ' → '[the man is sleeping] _C [though the company is talking with several prospects] _{SBAR} '
	TMP	VP < SBAR	'[the man is/was sleeping] _C SBAR' 'exports in October stood at \$ 5.29 billion, a mere 0.7% increase from a year earlier, [while imports increased sharply] _{SBAR} ' → '[the man was sleeping] _C [while imports increased sharply] _{SBAR} '
	-	VP < SBAR (direct speech)	'[the man says/said] _C SBAR' 'after the meeting, a Boeing spokeswoman said [a delivery date for the planes is still being worked out] _{SBAR} ' → '[the man said] _C [a delivery date for the planes is still being worked out] _{SBAR} '
	-	S < SBAR	'SBAR [, the man is sleeping] _C ' '[as word of the crime spree has spread] _{SBAR} [, many agents have started changing their open-door policies]' → '[as word of the crime spree has spread] _{SBAR} [, the man is sleeping] _C '
	-	NP < (SBAR << TO)	'[the man is writing a book] _C SBAR' 'Seoul also has instituted effective procedures [to aid these teams] _{SBAR} ' → '[the man is writing a book] _C [to aid these teams] _{SBAR} '
	-	NP < SBAR	NP _{rep}
	-	ADJP < (SBAR !< that)	'[the man knows that] _C SBAR' 'Mr Rey has been very careful since then to make sure [his moves are welcome] _{SBAR} ' → '[the man knows that] _C [his moves are welcome] _{SBAR} '
	-	ADJP < (SBAR < that)	'[the man knows] _C SBAR' 'this picture is about a middle-aged son who makes sure [that his delayed bond with his father will last] _{SBAR} ' → '[the man knows] _C [that his delayed bond with his father will last] _{SBAR} '
	-	PP → IN SBAR	'[the man knows] _C SBAR' 'depending on [how far down you go] _{SBAR} , it may be difficult to pay off that debt' → '[the man knows] _C [how far down you go] _{SBAR} '
	VP	TPC	all

Continued on next page

CAT	TAG	Environment	Context
	-	head = VBG	[the man is sleeping] _C VP 'the asbestos fiber is unusually resilient once it enters the lungs, with even brief exposures to it [causing symptoms that show up decades later] _{VP} ' → '[the man is sleeping] _C [causing symptoms that show up decades later] _{VP} '
	-	S < VP (subject SG)	[the man] _C VP ^a '[Pierre Vinken, 61 years old,] [will join the board] _{VP} ' → '[the man] _C [will join the board] _{VP} '
	-	S < VP (subject PL)	[the men] _C VP ^a '[four of the five surviving workers] [have asbestos-related diseases] _{VP} ' → '[the men] _C [have asbestos-related diseases] _{VP} '
		S < VP < VP	mimic original syntactic environment ^a
		S < VP < VP < VP	mimic original syntactic environment ^a
	-	ADJP → JJ (S < VP < VP)	'[the man wants to] _C VP' 'today's New England Journal of Medicine, a forum likely to [bring attention to the problem] _{VP} ' → '[the man wants to] _C [bring attention to the problem] _{VP} '
	-	ADJP → JJ (SBAR → WHNP (S < VP < VP))	'[the man wants to] _C VP' 'securities firms have scrambled to find new products that brokers find easy to [sell] _{VP} ' → '[the man wants to] _C [sell] _{VP} '
	-	NP → NP (head = NN) VP	'[the man,] _C VP' 'the new plant [located in Chinchon about 60 miles from Seoul] _{VP} will help ...' → '[the man,] _C [located in Chinchon about 60 miles from Seoul] _{VP} '
	-	NP → NP (head = NNS) VP	'[the men,] _C VP' 'the biggest reason earnings declined was a loss of production time and the increasing costs [associated with a temporary maintenance closing and expansion of an olefins plant] _{VP} ' → '[the men,] _C [associated with a temporary maintenance closing and expansion of an olefins plant] _{VP} '
	-	NP → NP (head = NNP) VP	'[john,] _C VP' 'GenCorp Inc , [hurt by a plant accident and other unexpected costs] _{VP} said it expects ...' → '[john,] _C [hurt by a plant accident and other unexpected costs] _{VP} '
	-	NP → NP (head = NNPS) VP	'[john and Alex,] _C VP' 'Georgia Gulf added 1 3/4 to 51 1/4 after NL Industries, [controlled by Dallas investor Harold Simmons] _{VP} offered . .' → '[john and Alex,] _C [controlled by Dallas investor Harold Simmons] _{VP} '
	-	NP → NP VP (remaining)	'[the man,] _C VP' 'Gary Hoffman, a Washington lawyer [specializing in intellectual-property cases] _{VP} , said the threat . .' → '[the man,] _C [specializing in intellectual-property cases] _{VP} '
NP _{rep}	-	X < NP (head = NN) Y	'[the car - a car - sugar] _C Y' ^b 'it received approval to sell [the first foldable silicone lens] _{NP} [available for cataract surgery] _{ADJP} ' → '[the car] _C [available for cataract surgery] _{ADJP} '

Continued on next page

^aBackoff code in case verbal pivot handling fails.

^bMimic definite article, indefinite article, mass noun environment.

CAT	TAG	Environment	Context
		'grants like Dupont and Maytag were on [the receiving end] _{NP} [of the message] _{PP} ' → '[the car] _C [of the message] _{PP} '	
		'Lorillard Inc, [the unit] _{NP} [that makes Kent cigarettes] _{SBAR} stopped using crocidolite ...' → '[the car] _C [that makes Kent cigarettes] _{SBAR} '	
		'Saudi Arabia has vowed to enact [a copyright law] _{NP} [compatible with international standards] _{ADJP} ' → '[a car] _C [compatible with international standards] _{ADJP} '	
		'shorter maturities are considered [a sign] _{NP} [of rising rates] _{PP} ' → '[a car] _C [of rising rates] _{PP} '	
		'it also would require the acquiring party to provide [all information] _{NP} [relevant to determining the intent of the acquisition] _{ADJP} ' → '[sugar] _C [relevant to determining the intent of the acquisition] _{ADJP} '	
	X < NP (head = NNS) Y		'[the cars] _C '
		'the Fed should guard against systemic risk, but not against [the risks] _{NP} [inherent in individual stocks] _{ADJP} ' → '[the cars] _C [inherent in individual stocks] _{ADJP} '	
		'systemwide sales, which includes [sales] _{NP} [at franchisee as well as company-owned stores] _{PP} ' → '[the cars] _C [at company-owned stores] _{PP} '	
	X < NP (head = NNP) Y		'[John] _C '
		'Mr. Rapanelli has said the government of [president Carlos Menem] _{NP} [who took office July 8] _{SBAR} ' → '[John] _C [who took office July 8] _{SBAR} '	
	X < NP (head = NNPS) Y		'[John and Alex] _C '
		'however, unlike [Messrs Graedel and Crutzen] [who are both pioneers in the study of atmospheric chemistry] _{SBAR} ' → '[John and Alex] _C [who are both pioneers in the study of atmospheric chemistry] _{SBAR} '	
	X < NP Y, default		'[the car] _C '
		'Rudolf Agnew, 55 years old and [former chairman] _{NP} [of Consolidated Gold Fields PLC] _{ADJP} ' → '[the car] _C [of Consolidated Gold Fields PLC] _{ADJP} '	

Table E.1. Static Context Templates per Category

Appendix F

Implementation: Class Diagram

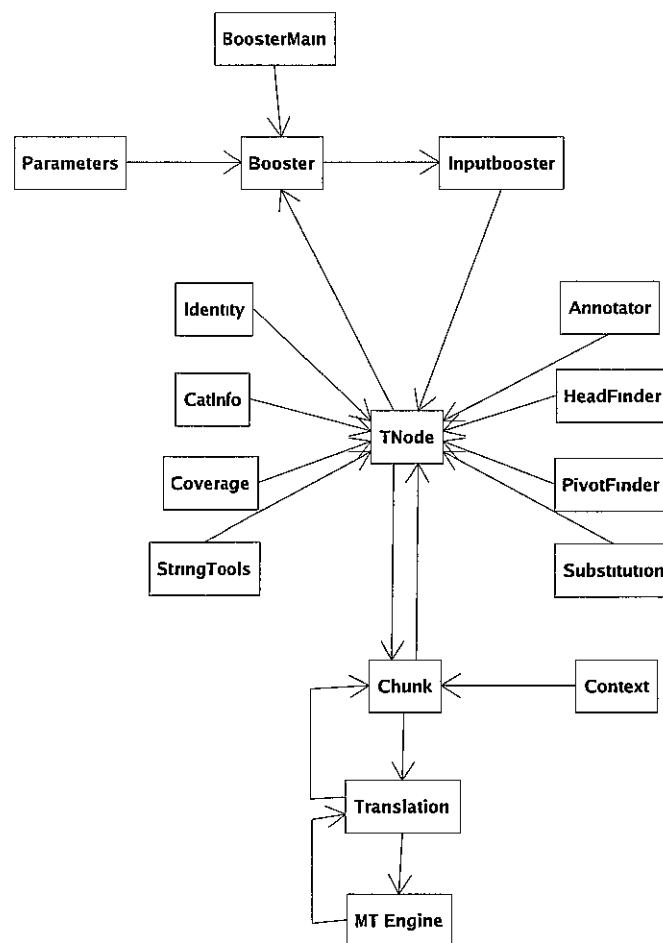


Figure F.1: Implementation of TransBooster Application (Java version J2SE 5.0) class diagram.

BoosterMain reads in the user options and starts the program execution.

Booster reads input data from a text file and coordinates the execution of the algorithm.

Parameters reads parameters from a text file and passes them on to Booster.

InputBooster converts a Penn-II Treebank tree into a collection of TransBooster Nodes or (*TNodes*).

TNode is a data structure representing a TransBooster Node and executes the main decomposition/recomposition algorithms at node level.

Identity contains information about TNode (head, adjunct or argument) and indicates its position in source and target

CatInfo contains the original Penn-II Treebank information for each TNode

Coverage produces several forms of lexical coverage of a TNode.

StringTools contains a number of useful String manipulating methods specific to Trans-Booster.

Annotator annotates each non-head TNode with argument/adjunct information

HeadFinder finds the head of a TNode.

PivotFinder finds the pivot of a TNode.

Substitution selects and stores static and dynamic Substitution Variables for satellites.

Context embeds a satellite in a static and dynamic context template.

Chunk is a data structure that stores pivot skeletons and satellites embedded in static/dynamic context templates. Chunk extracts the translation of each pivot/satellite from the translations of the embedded strings and passes the extracted translation to TNode

Translation interfaces Chunk with the baseline MT engine.

Table F.1 provides additional information about the amount of language-dependent code in the classes. Note that the vast majority of the language-dependent code is related to the source language, *not* to the target language. Only a limited number of methods regarding string retrieval in target are target language dependent. Column *Class* contains each of the relevant classes in the application. Column *Degree* specifies the degree to which the class is language-dependent ('none', 'low', 'medium' or 'high'). Column *Comments* contains further information on the language-dependent elements in each class.

Class	Degree	Comments
BoosterMain	none	Language independent.
Parameters	none	Language independent.
Booster	none	Language independent

Continued on next page

Class	Degree	Comments
InputBooster	low	Input = Penn-II tree. Source languages \neq English have to be parsed into Penn-like tree structures.
TNode	low	The main code for decomposition/recomposition is not language specific but depends on the correct identification and posterior processing of pivots, satellites and their SVs.
Annotator	high	The distinction between arguments and adjuncts is input language specific.
HeadFinder	high	Head-finding rules are input language specific.
Identity	none	Language independent.
CatInfo	none	Language independent.
Coverage	none	Language independent.
StringTools	medium	Most of the string manipulation methods in this class are language specific.
PivotFinder	high	The finding of a correct pivot is input language specific
Substitution	high	SSVs and DSVs are input language specific.
Chunk	none	This class relies on a correct identification of translation chunks and their contexts. The code itself is language independent.
Context	high	Static and Dynamic contexts are highly language specific.
Translation	none	Language independent.

Table F.1: Language-dependent vs Language-independent Elements in Trans-Booster

Bibliography

- Adriaens, G. and Caeyers, H. (1990). Het Automatisch Vertaalsysteem METAL: van Onderzoek tot Commercieel Produkt. *Ingénieur & Industrie*, pages 281–288.
- Akiba, Y., Imamura, K., and Sumita, E. (2001). Using Multiple Edit Distances to Automatically Rank Machine Translation Output. In *Machine Translation Summit VIII*, pages 15–20, Santiago de Compostela, Spain.
- Armstrong, S., Flanagan, M., Graham, Y., Groves, D., Mellebeek, B., Morrissey, S., Stroppa, N., and Way, A. (2006). MaTrEx: Machine Translation Using Examples. In *TC-STAR OpenLab on Speech Translation*, Trento, Italy. http://www.nclt.dcu.ie/mt/publications_06.html.
- Babych, B. and Hartley, A. (2004). Extending BLEU MT Evaluation Method with Frequency Weighting. In *Proceedings of the 42th Annual Meeting of the Association for Computational Linguistics (ACL)*, Barcelona, Spain.
- Banerjee, S. and Lavie, A. (2005). METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgements. In *Proceedings of Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the 43th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 65–72, Ann Arbor, MI.
- Bangalore, S., Bordel, G., and Riccardi, G. (2001). Computing Consensus Translation from Multiple Machine Translation Systems. In *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 351–354, Trento, Italy.

- Bar-Hillel, Y. (1960). A Demonstration of the Nonfeasibility of Fully Automatic High Quality Translation. *Advances in Computers*, 1:158-163. Appendix III
- Bennett, W. and Slocum, J. (1985). The LRC Machine Translation System. *Computational Linguistics*, 11:111-121.
- Bernth, A. and Gdaniec, C. (2001). MTranslatability. *Machine Translation*, 16(3):175-218.
- Bikel, D. M. (2002). Design of a Multi-lingual, Parallel-processing Statistical Parsing Engine In *Proceedings of the Human Language Technology Conference (HLT)*, pages 24-27, San Diego, CA.
- Brown, P. F., Pietra, S. D., Pietra, V. D., and Mercer, R. (1993) The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, pages 263-311.
- Brown, R. D. (1996) Example-Based Machine Translation in the Pangloss System. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pages 169-174, Copenhagen, Denmark.
- Burbank, A., Carpuat, M., Clark, S., Dreyer, M., Fox, P., Groves, D., Hall, K., Hearné, M., Melamed, D., Shen, Y., Way, A., Wellington, B., and Wu, D. (2005). Final Report of the Johns Hopkins Summer Workshop on Statistical Machine Translation by Parsing. In *JHU Workshop-2005*, Baltimore, MD.
- Burke, M (2006). *Automatic Treebank Annotation for the Acquisition of LFG Resources*. PhD thesis, School of Computing, Dublin City University, Dublin, Ireland.
- Cahill, A. (2004). *Parsing with Automatically Acquired, Wide-coverage, Robust, Probabilistic LFG Approximations*. PhD thesis, School of Computing, Dublin City University, Dublin, Ireland.
- Cahill, A., Burke, M., O'Donovan, R., van Genabith, J., and Way, A. (2004). Long-Distance Dependency Resolution in Automatically Acquired Wide-coverage PCFG-

- based LFG Approximations. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 319–326, Barcelona, Spain.
- Cahill, A. and van Genabith, J. (2002) TTS - a Treebank Tool. In *The Third International Conference on Language Resources and Evaluation (LREC)*, pages 1712–1717, Las Palmas de Gran Canaria, Spain. <http://research.computing.dcu.ie/~acahill/tts/>.
- Callison-Burch, C., Osborne, M., and Koehn, P. (2006). Re-evaluating the Role of Bleu in Machine Translation Research. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 249–256, Trento, Italy.
- Carl, M and Way, A. (2003). *Recent Advances in Example-Based Machine Translation*. Kluwer Academic Publishers, Dordrecht, The Netherlands.
- Chandioux, J. (1976). MÉTEO: Un système Opérationnel pour la Traduction Automatique des Bulletins Météorologiques Destinés au Grand Public. *META*, 21 127–133
- Chandioux, J. and Grimala, A. (1996). Specialized Machine Translation. In *Proceedings of the Second Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 206–211, Montreal, Canada
- Charniak, E. (1996). Tree-Bank Grammars. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI)*, pages 1031–1036, Menlo Park, CA.
- Charniak, E. (2000). A Maximum Entropy Inspired Parser. In *Proceedings of the First Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 132–139, Seattle, WA.
- Charniak, E., Knight, K., and Yamada, K. (2003). Syntax-based Language Models for Statistical Machine Translation. In *Machine Translation Summit IX*, pages 40–46, New Orleans, LO.
- Chiang, D. (2005). A Hierarchical Phrase-based Model for Statistical Machine Translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 263–270, Ann Arbor, MI.

- Cohen, A , Cousseau, P., Frederking, R., Grannes, D., Khanna, S., McNeilly, C., Nirenburg, S., Shell, P., and Waeltermann, D. (1993). *Translator's WorkStation User Document* Technical report, Center for Machine Translation, Carnegie Mellon University, Pittsburgh, PA.
- Collins, M. (1999). *Head-driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania, Philadelphia, PA
- Davison, A. and Hinkley, D. (1997). *Bootstrap Methods and their Application*. Cambridge University Press, Cambridge, UK.
- Deprez, F., Adriaens, G., Depoortere, B., and de Braekeleer, G. (1994). Experiences with Metal at the Belgian Ministry of the Interior. *META. Journal de traducteurs/Translators' Journal. Numéro Spécial, La Traduction et L'Interprétation dans la Belgique Multilingue*, 39(1):206–212.
- Doddington, G. (2002). Automatic Evaluation of MT Quality Using *n*-gram Co-occurrence Statistics. In *Proceedings of Human Language Technology Conference*, pages 128–132, San Diego, CA.
- Feng, D. and Doolittle, R. (1987). Progressive Sequence Alignment as a Prerequisite to Correct Phylogenetic Trees *Journal of Molecular Evolution*, 25:351–360.
- Fiscus, J. (1997). A Post-processing System to Yield Reduced Word Error Rates: Recognizer Output Voting Error Reduction (ROVER). In *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 238–245, Santa Barbara, CA
- Frederking, R., Cohen, A., Cousseau, P., Grannes, D , and Nirenburg, S. (1993). The Pangloss Mark I MAT System. In *Proceedings of the 6th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 468–468, Utrecht, The Netherlands.
- Frederking, R. and Nirenburg, S. (1994) Three Heads are Better than One. In *Proceedings of the Fourth Conference on Applied Natural Language Processing (ANLP)*, pages 95–100, Stuttgart, Germany.

- Gerber, L. and Hovy, E. (1998). Improving Translation Quality by Manipulating Sentence Length. In *Proceedings of the 3rd Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 448–460, Langhorne, PA.
- Goodman, K. and Nirenburg, S. (1991). *The KBMT Project: a Case Study in Knowledge-based Machine Translation*. Morgan Kaufman, San Mateo, CA.
- Gough, N. (2005). *Example-based Machine Translation Using the Marker Hypothesis*. PhD thesis, School of Computing, Dublin City University, Dublin, Ireland.
- Gough, N. and Way, A. (2004). Robust Large-Scale EBMT with Marker-Based Segmentation. In *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI)*, pages 95–104, Baltimore, MD.
- Green, T. (1979). The Necessity of Syntax Markers. Two Experiments with Artificial Languages. *Journal of Verbal Learning and Behavior*, 18:481–496.
- Groves, D. and Way, A. (2005). Hybrid Example Based SMT: the Best of Both Worlds. In *Workshop on Building and Using Parallel Texts at the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 183–190, Ann Arbor, MI.
- Groves, D. and Way, A. (2006). Hybridity in MT: Experiments on the Europarl Corpus. In *Proceedings of the 11th Conference of the European Association for Machine Translation*, pages 115–124, Oslo, Norway.
- Hockenmaier, J. (2003). *Data and models for Statistical Parsing with Combinatory Categorical Grammar*. PhD thesis, University of Edinburgh, Edinburgh, UK.
- Hovy, E. (1999). Toward Finely Differentiated Evaluation Metrics for Machine Translation. EAGLES Handbook, EAGLES Advisory Group. Pisa, Copenhagen, Geneva.
- Hovy, E., King, M., and Popescu-Belis, A. (2002). An Introduction to MT Evaluation. In *Handbook Workshop 'Machine Translation Evaluation: Human Evaluators Meet Automated Metrics' at the Third International Conference on Language Resources and Evaluation (LREC)*, pages 1–7, Las Palmas de Gran Canaria, Spain.

- Hutchins, J., Hartmann, W., and Ito, E. (2006). Compendium of Translation Software Technical report, European Association for Machine Translation (EAMT). <http://www.eamt.org/compendium.html>.
- Hutchins, W and Somers, H. (1992). *An Introduction to Machine Translation*. Academic Press Ltd
- Jayaraman, S. and Lavie, A. (2005). Multi-Engine Machine Translation Guided by Explicit Word Matching. In *Proceedings of the 10th Conference of the European Association for Machine Translation (EAMT)*, pages 143–152, Budapest, Hungary.
- Kaplan, R. and Bresnan, J (1982). Lexical Functional Grammar, a Formal System for Grammatical Representation. In Bresnan, J., editor, *The Mental Representation of Grammatical Relations*, pages 173–281. MIT Press, Cambridge, MA.
- King, T. H., Crouch, R., Riezler, S., Dalrymple, M., and Kaplan, R. M. (2003) The PARC 700 Dependency Bank. In *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora, held at the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 1–8, Budapest, Hungary
- Kneser, R and Ney, H. (1995). Improved Backing-off for m -gram Language Modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 181–184, Detroit, MI.
- Koehn, P (2004). Pharaoh: A Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. In *Proceedings of the 6th Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 115–124, Georgetown University, Washington DC.
- Koehn, P. (2005). Europarl: a Parallel Corpus for Evaluation of Machine Translation In *Machine Translation Summit X*, pages 79–86, Phuket, Thailand.
- Koehn, P. and Monz, C. (2006). Manual and Automatic Evaluation of Machine Translation between European Languages In *Proceedings on the Workshop on Statistical Machine Translation at the joint Conference Human Language Technology - North American*

- Chapter of the Association for Computational Linguistics (HLT-NAACL), pages 102–121, New York, NY.
- Koehn, P., Och, F., and Marcu, D. (2003). Statistical Phrase-based Translation. In *Proceedings of the joint Conference Human Language Technology - North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, pages 127–133, Edmonton, Canada.
- Krings, H. (2001). *Repairing Texts. Empirical Investigations of Machine Translation Post-Editing Processes*. Kent State University Press, Kent, OH
- Landsbergen, J. (1989). The Rosetta Project. In *Machine Translation Summit II*, pages 82–87, Munich, Germany.
- Larkey, L. and Croft, B. (1996). Combining Classifiers in Text Categorization. In *Proceedings of the 19th Annual International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 289–297, Zurich, Switzerland.
- Lehrndorfer, A. and Schachtl, S. (1998). TR09 Controlled Siemens Documentary German and TopTrans In *TC Forum*, number 3.
- Lepage, Y. (2005). The Purest Ever Built EBMT System: no Variables, no Templates, no Training, Examples, just Examples, only Examples. In *Proceedings of Second Workshop on Example-Based Machine Translation at Machine Translation Summit X*, pages 81–90, Phuket, Thailand.
- Leusch, G., Ueffing, N., and Ney, H. (2003). A Novel String-to-String Distance Measure with Applications to Machine Translation Evaluation. In *Machine Translation Summit IX*, pages 240–247, New Orleans, LO.
- Levenshtein (1965). Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Doklady Akademii Nauk SSR*, 163(4), pages 845–848.
- Li, A. (2005). Results of the 2005 NIST Machine Translation Evaluation. Technical report.
- Lin, C. and Och, F. (2004a). Automatic Evaluation of Machine Translation Quality Using Longest Common Subsequence and Skip-Bigram Statistics. In *Proceedings of the 42nd*

- Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 605–612, Barcelona, Spain.
- Lin, C. and Och, F. (2004b). Orange: a Method for Evaluating Automatic Evaluation Metrics for Machine Translation. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, pages 501–507, Geneva, Switzerland.
- Locke, W. and Booth, A. (1955). *Machine Translation of Languages*. MIT Press, Cambridge, MA.
- Maas, H. (1987). The MT System SUSY. In King, M., editor, *Machine Translation Today: the state-of-the-art*, pages 209–246. Edinburgh University Press, Edinburgh, UK.
- Magerman, D. (1995). Statistical Decision-Tree Models for Parsing. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 276–283, Cambridge, MA.
- Marcu, D. and Wong, W. (2002). A Phrase-based, Joint Probability Model for Statistical Machine Translation. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 133–139.
- Marcus, M., Kim, G., Marcinkiewicz, M., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., and Schasberger, B. (1994). The Penn Treebank: Annotating Predicate Argument Structure. In *Proceedings of the ARPA Human Language Technology Workshop*, pages 114–119, Plainsboro, NJ.
- Matusov, E., Ueffing, N., and Ney, H. (2006). Computing Consensus Translation from Multiple Machine Translation Systems using Enhanced Hypotheses Alignment. In *11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 33–40, Trento, Italy.
- Mellebeek, B., Khasin, A., van Genabith, J., and Way, A. (2005a). TransBooster Boosting the Performance of Wide-coverage Machine Translation Systems. In *Proceedings of the 10th Annual Conference of the European Association for Machine Translation (EAMT)*, pages 189–197, Budapest, Hungary.

- Mellebeek, B., Khasin, A., Owczarzak, K., van Genabith, J., and Way, A. (2005b). Improving Online Machine Translation Systems. In *Machine Translation Summit X*, pages 290–297, Phuket, Thailand.
- Mellebeek, B., Owczarzak, K., Groves, D., van Genabith, J., and Way, A. (2006a). A Syntactic Skeleton for Statistical Machine Translation. In *Proceedings of the 11th Annual Conference of the European Association for Machine Translation (EAMT)*, pages 195–202, Oslo, Norway.
- Mellebeek, B., Owczarzak, K., van Genabith, J., and Way, A. (2006b). Multi-Engine Machine Translation by Recursive Sentence Decomposition. In *Proceedings of the 7th Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 110–118, Boston, MA.
- Menezes, A. and Quirk, C. (2005). Dependency Treelet Translation: the Convergence of Statistical and Example-Based Machine Translation. In *Proceedings of Second Workshop on Example-Based Machine Translation at Machine Translation Summit X*, pages 99–108, Phuket, Thailand.
- Menezes, A. and Richardson, S. (2001). A Best-first Alignment Algorithm for Automatic Extraction of Transfer Mappings from Bilingual Corpora. In *Proceedings of the Workshop on Data-driven Machine Translation in conjunction with the 39th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1–8, Toulouse, France.
- Nagao, M. (1984). A Framework of a Mechanical Translation between Japanese and English by Analogy Principle. In Elithorn, A. and Banerji, R., editors, *Artificial and Human Intelligence*, pages 173–180. North-Holland, Amsterdam, The Netherlands.
- Nießen, S., Och, F., Leusch, G., and Ney, H. (2000). An Evaluation Tool for Machine Translation: Fast Evaluation for MT Research. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC)*, pages 39–45, Athens, Greece.
- Nirenburg, S., Domashnev, C., and Grannes, D. (1993). Two Approaches to Matching in Example-Based Machine Translation. In *Proceedings of the 5th International Conference*

- on *Theoretical and Methodological Issues in Machine Translation (TMI)*, pages 47–57, Kyoto, Japan.
- Nomoto, T. (2004). Multi-Engine Machine Translation with Voted Language Model In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 494–501, Barcelona, Spain.
- Nyberg, E. and Mitamura, T. (1992). The KANT System: Fast, Accurate, High-quality Translation in Practical Domains In *Proceedings of the 15th International Conference on Computational Linguistics (COLING)*, pages 1069–1073, Nantes, France.
- O’Brien, S. (2003) Controlling Controlled English an Analysis of Several Controlled Language Rule Sets. In *8th International Workshop of the European Association for Machine Translation*, Dublin, Ireland.
- Och, F. and Ney, H. (2002). Discriminative Training and Maximum-Entropy Models for Statistical Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 295–302, Philadelphia, PA.
- Och, F. and Ney, H. (2003). A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29:19–51.
- O’Donovan, R (2006). *Large Scale Multilingual Lexical Extraction*. PhD thesis, School of Computing, Dublin City University, Dublin, Ireland
- Owczarzak, K., Mellebeek, B., Groves, D., van Genabith, J., and Way, A. (2006). Wrapper Syntax for Example-based Machine Translation. In *Proceedings of the 7th Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 148–155, Boston, MA
- Papineni, K., Roukos, S , Ward, T., and Zhu, W. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318, Philadelphia, PA.
- Pito, R. (1993) Documentation for tgrep. Technical report, LDC, University of Pennsylvania

- nia, Philadelphia, PA. <http://www ldc.upenn.edu/ldc/online/treebank/README.long>.
- Roth, D. and Zelenko, D (1998) Part-of-speech Tagging Using a Network of Linear Separators In *Proceedings of the 17th International Conference on Computational Linguistics (COLING) and 36th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1136–1142, Montreal, Canada
- Rychtycky, N. (2002). An Assessment of Machine Translation for Vehicle Assembly Process Planning at Ford Motor Company In *Proceedings of the 5th Biennial Conference of the Association for Machine Translation in the Americas (AMTA)· From Research to Real Users*, pages 207–215, Tiburon, CA.
- Senellart, J., Dienes, P., and Váradi, T. (2001). New Generation SYSTRAN Translation System. In *Machine Translation Summit VIII*, pages 311–316, Santiago de Compostela, Spain
- Somers, H. (2003). An Overview of EBMT. In Carl, M. and Way, A., editors, *Recent Advances in Example-based Machine Translation*, pages 3–57. Kluwer Academic Publishers, Dordrecht, The Netherlands
- Steedman, M. (1996) *Surface Structure and Interpretation*. MIT Press, Cambridge, MA.
- Steedman, M. (2000). *The Syntactic Process*. MIT Press, Cambridge, MA.
- Stolcke, A. (2002). SRILM - an Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 901–904, Denver, CO.
- Thurmair, G. (1992). METAL: Computer Integrated Translation. Technical report, Siemens-Nixdorf, Munich, Germany. Internal METAL documentation.
- Trujillo, A. (1999). *Translation Engines: Techniques for Machine Translation* Springer-Verlag, London, UK.
- Turian, J., Shen, L., and Melamed, D. (2003). Evaluation of Machine Translation and its Evaluation. In *Machine Translation Summit IX*, pages 386–393, New Orleans, LO.

- van Rijsbergen, C. (1979). *Information Retrieval*. Butterworths, London, UK.
- van Zaanen, M. and Somers, H (2005). DEMOCRAT: Deciding between Multiple Outputs Created by Automatic Translation. In *Machine Translation Summit X*, pages 173–180, Phuket, Thailand.
- White, J. and Connell, T. O. (1994). The ARPA MT Evaluation Methodologies: Evolution, Lessons and Future Approaches. In *Proceedings of the First Conference of the Association for Machine Translation in the Americas (AMTA)*, pages 193–205, Columbia, MD.
- Wu, D. (2005). MT Model Space: Statistical vs. Compositional vs. Example-Based Machine Translation. *Machine Translation*, (19):213–227.
- Wu, D and Wong, H (1998) Machine Translation with a Stochastic Grammatical Channel. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (ACL-COLING)*, pages 1408–1415, Montreal, Canada.
- Xia, F. and McCord, M. (2004). Improving a Statistical MT System with Automatically Learned Rewrite Patterns. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, pages 508–514, Geneva, Switzerland.
- Yamada, K. and Knight, K. (2001). A Syntax-based Statistical Translation Model. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 523–530, Toulouse, France.
- Zhang, Y and Vogel, S. (2004). Measuring Confidence Intervals for the Machine Translation Evaluation Metrics. In *Proceedings of the 10th Conference on Theoretical and Methodological Issues in Machine Translation (TMI)*, pages 85–94, Baltimore, MD.
- Zhang, Y , Vogel, S., and Waibel, A. (2004). Interpreting BLEU/NIST Scores: How much improvement do we need to have a better system? In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC)*, pages 2051–2054, Lisbon, Portugal.