Technical Report

UCAM-CL-TR-913
ISSN 1476-2986

Number 913

UNIVERSITY OF CAMBRIDGE

Computer Laboratory

Hierarchical statistical semantic translation and realization

Matic Horvat

October 2017

This technical report is based on a dissertation submitted March 2017 by the author for the degree of Doctor of Philosophy to the University of Cambridge, Churchill College.

ISSN 1476-2986

Hierarchical statistical semantic translation and realization

Matic Horvat

Summary

Statistical machine translation (SMT) approaches extract translation knowledge automatically from parallel corpora. They additionally take advantage of monolingual text for target-side language modelling. Syntax-based SMT approaches also incorporate knowledge of source and/or target syntax by taking advantage of monolingual grammars induced from treebanks, and semantics-based SMT approaches use knowledge of source and/or target semantics in various forms. However, there has been very little research on incorporating the considerable monolingual knowledge encoded in deep, hand-built grammars into statistical machine translation. Since deep grammars can produce semantic representations, such an approach could be used for realization as well as MT.

In this thesis I present a hybrid approach combining some of the knowledge in a deep hand-built grammar, the English Resource Grammar (ERG), with a statistical machine translation approach. The ERG is used to parse the source sentences to obtain Dependency Minimal Recursion Semantics (DMRS) representations. DMRS representations are subsequently transformed to a form more appropriate for SMT, giving a parallel corpus with transformed DMRS on the source side and aligned strings on the target side. The SMT approach is based on hierarchical phrase-based translation (Hiero). I adapt the Hiero synchronous context-free grammar (SCFG) to comprise graph-to-string rules. DMRS graph-to-string SCFG is extracted from the parallel corpus and used in decoding to transform an input DMRS graph into a target string either for machine translation or for realization.

I demonstrate the potential of the approach for large-scale machine translation by evaluating it on the WMT15 English-German translation task. Although the approach does not improve on a state-of-the-art Hiero implementation, a manual investigation reveals some strengths and future directions for improvement. In addition to machine translation, I apply the approach to the MRS realization task. The approach produces realizations of high quality, but its main strength lies in its robustness. Unlike the established MRS realization approach using the ERG, the approach proposed in this thesis is able to realize representations that do not correspond perfectly to ERG semantic output, which will naturally occur in practical realization tasks. I demonstrate this in three contexts, by realizing representations derived from sentence compression, from robust parsing, and from the transfer-phase of an existing MT system.

In summary, the main contributions of this thesis are a novel architecture combining a statistical machine translation approach with a deep hand-built grammar and a demonstration of its practical usefulness as a large-scale machine translation system and a robust realization alternative to the established MRS realization approach.

Acknowledgments

I am grateful to many people for their support and contributions to this thesis. Above all, I will be forever indebted to my PhD supervisors Ann Copestake and Bill Byrne for their mentorship and guidance and the knowledge they imparted over the course of my time in Cambridge. In particular, I would like to thank Ann for her unwavering support, and Bill, for his generosity to have me as his student. Some students get lucky to have one great supervisor. I had two.

I would like to thank Juan Pino and Rory Waite for teaching me the ropes of building and using statistical machine translation systems, and Gonzalo Iglesias and Adria de Gispert for discussions that helped shape the approach and its implementation presented in this thesis. They and others are the authors of the HiFST SMT decoder, which was a significant source of inspiration for my own work. I would furthermore like to thank Eva Hasler for the useful discussions and in particular for sharing her DMRS-based sentence compression code.

I am grateful to Guy Emerson, Alex Kuhnle, and Ewa Muszyńska for the constructive discussions and their help with evaluating German translation outputs. My gratitude also goes to other members of the NLIP group, for the discussions, learning opportunities, and sharing the burden of a PhD.

I would like to thank Woodley Packard for his help and advice on using the ERG and the ACE parser, Mike Goodman for his help and advice on using the pyDelphin library, Francis Bond for his help and advice on setting up the Japanese-English transfer-based MT system, and Dan Flickinger for the helpful discussions and constructive feedback, and in particular for the treebanking of a gold dataset. I am also grateful to the other members of the DELPH-IN community, whose efforts I built my work on.

I would like to thank my examiners, Francis Bond and Paula Buttery, for the insightful discussion of the thesis and helpful comments and suggestions that made it better than it was.

During my PhD I got to spent an educational summer at the Information Sciences Institute, for which I would like to thank everyone at ISI and in particular Kevin Knight, Daniel Marcu, and Jon May, as well as fellow interns Eunsol Choi, Julian Schamper, and Allen Schmaltz. Likewise, I would like to thank my friends and colleagues at Cytora for giving me an opportunity to apply my knowledge.

The work presented in this thesis could not have been accomplished without the resources generously provided by the University of Cambridge Computer Laboratory and Department of Engineering. I would also like to express my sincere gratitude to Churchill College and Qualcomm for providing the financial support that allowed me to pursue a PhD.

Furthermore, Churchill College has become a second home and has given me countless memories that will stay with me for the rest of my life. Thank you to my friends, both in Cambridge and in Ljubljana, for their encouragement during these years. In particular, a big thank you goes to Sara Saheb Kashaf for her relentless support and help with putting the finishing touches to the thesis. Last but definitely not least, I would like to thank my family and in particular my mom and dad for their encouragement, support, and everything they've done for me.

Matic Horvat, Cambridge, March 2017

Contents

Chapter 1

Introduction

1.1 Motivation

Statistical machine translation (SMT) approaches starting with the seminal work by Brown et al. (1990) transformed the field of machine translation. Whereas preceding approaches used translation knowledge encoded by humans, statistical approaches obtain translation knowledge automatically from sentence-aligned parallel corpora. The benefit of using parallel corpora is the reduced cost of acquiring translation knowledge, provided that a parallel corpus of sufficient size for a given language pair exists. In comparison, transfer-based machine translation approaches take advantage of the translation knowledge encoded by humans as well as the considerable monolingual knowledge in each of the languages. In contrast to statistical approaches, the translation and monolingual knowledge encoded by humans is typically more precise but sparser, yielding high quality translations at the expense of coverage (in particular, lexical coverage) and robustness.

To take a specific example, the monolingual knowledge in the transfer-based machine translation approaches by Oepen et al. (2004a) and Bond et al. (2005) is encoded in a broad-coverage precision Head-Driven Phrase Structure Grammars (HPSGs). A broad-coverage computational HPSG, such as the English Resource Grammar (Flickinger, 2000), is developed by a linguist by (1) using a corpus to discover phenomena yet untreated by the HPSG, and (2) extending the HPSG by encoding a generalization of the phenomena using their linguistic intuitions and consulting linguistic literature (Baldwin et al., 2005). In comparison, a probabilistic context-free grammar (PCFG) is estimated from a human-annotated treebank, consisting of sentences annotated with corresponding syntactic trees according to annotation guidelines. Because of this, an HPSG contains knowledge (e.g., in the form of generalizations) which cannot necessarily be extracted from a treebank automatically. Typically, a broad-coverage computational HPSG aims to be a high-precision grammar, producing high quality analyses at the expense of coverage and robustness. An example of a broad-coverage high-precision HPSG is the English Resource Grammar (ERG). The ERG is a bidirectional grammar, which allows it to be used both for producing analyses of input sentences (i.e., parsing) as well as producing sentences from an input meaning representation (i.e., realization). The meaning representation framework used by the ERG is Minimal Recursion Semantics (Copestake et al., 2005).

In order to achieve the best possible translation performance, machine translation systems use as much knowledge as possible. In the case of statistical machine translation, the larger the size of the parallel corpus, the more knowledge can be extracted and the better performance the SMT system can achieve. However, since performance of an SMT system only grows logarithmically with increasing amount of parallel data (Och, 2005), it is difficult to continue improving it by incorporating more knowledge in the form of parallel text beyond a certain point. Because of this, knowledge from monolingual resources is incorporated into the statistical machine translation system. A simple but crucial source of knowledge for SMT approaches is the target language model, which is based on monolingual text in the target language. Syntax-based statistical machine translation approaches (see Section 2.1) also incorporate knowledge of source and/or target syntax by taking advantage of monolingual grammars induced from treebanks (such as the Penn Treebank (Marcus et al., 1993)). Following the success of syntax-based SMT approaches, semantics-based SMT approaches incorporate knowledge of source and/or target semantics, including lexical semantics, predicate-argument structure, and semantic role labelling information (see Section 2.2). However, no attempt has been made so far to incorporate the considerable knowledge encoded in deep, hand-built grammars into statistical machine translation.[1]

In this thesis I propose a new type of hybrid architecture which combines broad-coverage grammar-based semantics with statistical machine translation approaches. In addition to the knowledge encoded in a sentence-aligned parallel corpus (as used in SMT), the proposed approach takes advantage of the linguistic knowledge encoded in the high-precision English Resource Grammar. Specifically, the knowledge is incorporated into the source side of a parallel corpus by parsing the source sentences using the ERG to obtain their semantic representations. The statistical approach is based on hierarchical phrase-based translation (Chiang, 2005, 2007), extended to take advantage of source-side semantic representations encoded as graphs.

The source-side semantic representations encode aspects of lexical semantics and are in many respects similar to a predicate-argument structure. In previous work, both types of semantics were integrated into existing SMT approaches. However, the approach proposed in this thesis goes beyond integrating semantics into an existing SMT approach. It proposes to translate the semantic structures themselves. In that respect, it is similar to syntax-based tree-to-string translation approach of Liu et al. (2006) and dependency-to-string approaches of Xiong et al. (2007) and Li et al. (2015, 2016). It is no coincidence that it builds on some of the machinery commonly used in syntax-based SMT. A related semantics-based SMT approach by Zhai et al. (2013) maps predicate-argument structures to the target side to provide a skeleton for the remaining translations. Perhaps the most related approach in semantics-based SMT is the approach by Jones et al. (2012), who present a statistical system translating Abstract Meaning Representations (Banarescu et al., 2013), represented as directed acyclic graphs, using synchronous hyperedge replacement grammars. I review the related machine translation approaches in Chapter 2.

A strength of the statistical machine translation approach presented in this thesis is its robustness. In contrast, robustness and coverage have typically been a weakness of approaches using the ERG and other hand-built deep grammars. The lack of robustness is in large part a consequence of the high precision requirement. While the high-precision outputs benefit the many applications of deep grammars, their lack of robustness limits

[1]The approach by Graham et al. (2009) takes advantage of similar knowledge encoded in the Lexical Functional Grammar, but in a transfer-based machine translation approach with a statistical transfer component. I discuss their work in more detail in related work, Section 2.3.

its usefulness for others. In order to address part of the robustness problem, a robust parsing approach was recently proposed by Packard and Flickinger (2017) following the work of Zhang and Krieger (2011). The robust parsing approach is capable of achieving much higher parsing coverage than the established approach, including parsing of ungrammatical inputs. The approach is estimated on ERG parse trees and is therefore more tightly coupled with the existing parsing approach compared to a standalone statistical back-off approach. So far, no such solution existed for the reverse problem of realization.

In this thesis I use the same statistical machine translation architecture as for machine translation for robust English Resource Grammar realization. In a similar vein to the robust parsing approach, the robust realization approach using SMT is capable of realizing flawed and ill-formed inputs, while taking advantage of the existing grammar knowledge, as opposed to building a completely separate system.

In order to perform as well as possible in practical applications, a statistical machine translation system must be capable of taking advantage of all available parallel training data, in addition to incorporating new monolingual knowledge.[2] In order to achieve that, the approach proposed in this thesis is accompanied with an efficient implementation that allows it to take advantage of millions of parallel training examples and be evaluated on real-world inputs for both translation and realization applications.

To summarize, in this thesis I investigate two questions: (1) *Can the hand-built deep grammar based approaches and the statistical machine translation based approaches be combined in a single, large-scale system?*; and (2) *Is the proposed approach promising for the tasks of machine translation and realization?*.

1.2 Hierarchical statistical semantic translation/realization

Statistical machine translation approaches extract translation knowledge from sentence-aligned parallel data and subsequently apply it to previously unseen sentences in the source language to produce translations in the target language. In the approach proposed in this thesis, the source-side sentences are parsed with the English Resource Grammar in order to take advantage of the knowledge of the source language encoded in the deep grammar. Using the resulting source-side semantics in a statistical machine translation approach requires extending the existing statistical machine translation machinery. These extensions constitute the approach proposed in this thesis (illustrated in a diagram later in this chapter in Figure 1.1). In this section I describe my approach and motivation for it in more detail.

The English Resource Grammar constructs a meaning representation, the Minimal Recursion Semantics (MRS) representation, compositionally during parsing. The MRS representation aims to include all semantically related information that can be derived from syntax and morphology. The MRS representations have been demonstrated to be useful in machine translation (Oepen et al., 2004a; Nichols et al., 2007; Bond et al., 2005, 2011). However, a more convenient semantic representation for the statistical machine translation approaches considered in this thesis is a Dependency Minimal Recursion Semantics

[2]The alternative to such a large-scale system is a toy system with limited practical utility.

(DMRS) representation (Copestake, 2009). DMRS representations are inter-convertible with MRS, but encode the semantics of a sentence as a directed acyclic graph instead of a flat semantic representation.

Not all information encoded in a DMRS graph is useful for its various applications. In fact, some information may even harm its utility for a particular application. For instance, some types of nodes in a DMRS graph do not contribute useful translation information and are therefore redundant for the machine translation task considered in this thesis. If such nodes are kept in a statistical translation approach, they increase the noise in the parallel data while consuming extraneous computing resources. Because of this, I introduce three methods for transforming DMRS graphs in order to make them more suitable for use with the statistical machine translation approach, and translation and realization tasks considered in this thesis.

The translation knowledge in statistical approaches is extracted from parallel sentences. This requires an *alignment* between the words in a source sentence and words in a target sentence. The *word alignment* is commonly produced using an unsupervised statistical approach. Since the source sides considered in this thesis are DMRS graphs, their alignment to target sentence words requires some adaptation. The method proposed in this thesis extends the source-side alignment information derived during ERG parsing, and combines them with the word alignments produced by an unsupervised statistical approach. Together, the DMRS graph modelling and training example alignment methods are described in Chapter 3.

The aligned parallel corpus is used by a statistical machine translation approach to extract translation knowledge. My approach is based on hierarchical phrase-based translation (Hiero). In Hiero, translation knowledge is encoded as a *synchronous context-free grammar* (SCFG). A Hiero SCFG is extracted from word-aligned pairs of sentences (see Section 4.1). However, since the source side is a DMRS graph in my approach, I extended the Hiero SCFG formalism to comprise graph-to-string rewrite rules. The extension of SCFG to source-side graphs also requires a novel rule extraction algorithm, which extracts a set of rules from a single aligned DMRS graph - sentence pair.

In order to obtain a grammar that is capable of producing high quality translations or realizations for a broad set of input DMRS graphs, rules are extracted from millions of training examples. The extracted rules therefore need to be aggregated into the SCFG. Aggregating graph-to-string rules requires addressing the graph isomorphism problem, which I address with a heuristic solution to the related problem of graph canonization. Since the resulting SCFG encodes the translation knowledge, it is interesting to investigate its structure. Particularly compelling is the analysis that compares the differences in the extracted SCFGs for translation and realization tasks. The grammar extraction and its subsequent analysis are described in Chapter 4.

In hierarchical phrase-based translation, a previously unseen sentence is translated using the translation knowledge encoded in an SCFG in a process called *decoding* (see Section 5.1). The input sentence is parsed using the CYK algorithm and the resulting space is explored to find the highest scoring translation. However, since the source sentence is already parsed with the ERG to produce the DMRS graph in the proposed approach, a different decoding strategy is needed. Instead, I propose a novel rule application algorithm which determines the set of SCFG rules that can be applied to the input DMRS graph in order to translate it to the target string.

Each rule translates only a part of the input DMRS graph and rules are applied hierarchically to translate larger parts of the graph until the whole graph is translated. However, as there exist many ways of translating a sequence of words, there exist many ways of translating a part of a DMRS graph. Consequently, the competing rules produce competing translation hypotheses. I encode the competing hypotheses in a *hypothesis space* represented as a Finite State Acceptor (FSA), adapting the ideas of de Gispert et al. (2010) to graph-to-string decoding. In order to choose the best translation, I score the individual hypothesis with a log-linear model and the highest scoring hypothesis is chosen. Decoding real-world inputs using an SCFG extracted from millions of training examples is a resource intensive procedure. Consequently, I analyse the rule application algorithm and decoder performance for both tasks considered in this thesis. The decoding part of the approach and its performance analysis are described in Chapter 5.

In summary, the hybrid architecture combining statistical machine translation with broad-coverage semantics presented in this thesis consists of three parts: (1) approaches and algorithms concerned with modelling of input DMRS graphs and parallel training examples (i.e., *semantic data modelling*, Chapter 3); (2) approaches and algorithms concerned with extracting a synchronous-context free grammar from a corpus of aligned parallel examples (i.e., *rule extraction*, Chapter 4); and (3) approaches and algorithms concerned with transforming a previously unseen DMRS graph into a target sentence using the extracted synchronous context-free grammar (i.e., *decoding*, Chapter 5).

1.3 Machine translation

Machine translation is the task of translating a sentence in the source language to the sentence in the target language while preserving the meaning conveyed by the source sentence in the target sentence without direct human assistance.

The aim of the approach introduced in Section 1.2 and described in detail in Chapters 3, 4, and 5 is to take advantage of the monolingual knowledge encoded in the deep English Resource Grammar by combining it with a statistical machine translation approach. In order to investigate its potential as a large-scale translation system, I evaluate my approach on the large-scale WMT-15 English-German translation task. The training dataset consists of more than 4 million parallel sentences. I compare my approach to HiFST (Blackwood et al., 2016), an established state-of-the-art implementation of hierarchical phrase-based translation. The translation evaluation is described in Chapter 6.

Extracted translation knowledge, regardless of the parallel corpus size, is not exhaustive for a broad-domain translation task due to the generative nature of language. A problem occurs when a machine translation system encounters an input which it cannot translate (i.e., due to the lack of required translation knowledge).[3] A common solution to translating such inputs is to map them directly from the source language to the target language. In Chapter 6, I introduce four types of *non-grammar rules* that improve the robustness of my approach by taking advantage of the information encoded in the source-side DMRS graphs.

[3]Trivial examples of such inputs include made-up names (e.g., *Bellywinklestein*) and large numbers (e.g., *5,623,745*), neither of which are likely to have occured in the parallel corpus.

1.4 Realization

Natural language generation (NLG) is a sub-field of *natural language processing* (NLP) concerned with producing utterances in a natural language from a computer-internal representation. Since natural language generation is not as standardised as machine translation, I discuss it here in more detail.

The natural language generation task is commonly decomposed into *content planning* and *surface realization*. While content planning is concerned with *what* needs to be communicated, surface realization is concerned with *how* the content is to be communicated. Therefore, surface realization (also referred to as just *realization*) is the task of creating a fluent grammatical sentence from a meaning representation so that the sentence preserves the meaning encoded by the representation.

This interpretation of the surface realization task assumes that the meaning representation is perfect and is sometimes referred to as *tactical generation*. In contrast, the *regeneration* interpretation of the realization task accepts that a meaning representation is a potentially flawed representation of the original sentence (or another object), because it was obtained via an imperfect process, such as automatic parsing. Whereas in tactical generation the main goal is to preserve the meaning of the input meaning representation, the main goal in regeneration is to preserve the meaning conveyed by the original sentence. Additionally, coverage and robustness are paramount in regeneration, since a flawed or broken representation is an acceptable input.

A common way of representing the content to be communicated in an utterance is with a *meaning representation*. Meaning representations differ greatly across the NLG literature in terms of their structural properties and the degree to which they pre-determine their natural language realization. Perhaps the shallowest meaning representation is a bag-of-words. The associated realization task is referred to as string regeneration and corresponds to arranging the bag-of-words into a fluent sentence. Approaches for addressing the string regeneration problem span a wide range of techniques, for instance CCG parsing (Zhang and Clark, 2011), phrase-based machine translation (de Gispert et al., 2014), and graph-based modelling (Horvat and Byrne, 2014). On the other side of the spectrum, Abstract Meaning Representation (AMR) is a representation which only loosely constrains its natural language realization. For example, a recent approach by Konstas et al. (2017) uses neural sequence-to-sequence models for both parsing and realization. The related realization approaches are reviewed in Chapter 2.

The meaning representation considered in this thesis is Dependency Minimal Recursion Semantics (DMRS). DMRS is a semantic representation grounded in predicate calculus (see Section 3.1 for an introduction). However, since the representation is obtained compositionally via HPSG parsing, it remains somewhat tied to sentence syntax. Because of this, a DMRS representation pre-determines its natural language realizations more than, for instance, AMR does. The established approach to realization of DMRS representations uses chart generation with a bidirectional grammar (Carroll et al., 1999), such as the ERG, and a stochastic realization reranking component (Velldal and Oepen, 2005). Due to its reliance on the grammar, the established approach tends to produce grammatical and fluent outputs that are guaranteed to be meaning preserving with respect to the meaning representation. However, when realizing from meaning representations created or modified by a system other than the bidirectional grammar, the established approach is inevitably brittle. Consequently, the established approach to MRS realization is well-suited for tactical generation, but the lack of robustness means that it is less well-suited for regeneration.

The statistical machine translation approach presented in this thesis has a complementary set of strengths and weaknesses. Although it aims to produce grammatical and fluent outputs of comparable quality to the established approach, it cannot hope to surpass it. On the other hand, the statistical methods make it capable of realizing flawed or broken meaning representations. This makes the proposed approach particularly well-suited for the regeneration interpretation of the realization task and as a robust alternative to the established approach. As discussed in Section 1.1, the robust realization approach fills a void in the literature and is comparable in its purpose and objectives to the recently proposed robust parsing approaches (Packard and Flickinger, 2017).

I evaluate the hybrid approach under both the tactical generation and regeneration interpretations of the realization task. I compare it to the established approach to MRS realization by mirroring the machine translation evaluation methodology. I define and measure additional metrics that further elucidate the output quality of the proposed approach in terms of meaning preservation and grammaticality. In order to demonstrate the robustness of the proposed approach I set up three tasks in which the meaning representations are flawed or broken in various ways: (1) realization of heavily modified and compressed DMRS graphs for the sentence simplification task; (2) realization of representations produced by the robust parser; and (3) realization of transferred representations produced by a transfer-based machine translation system. The realization task evaluation is described in Chapter 7.

1.5 System overview

Machine translation systems are complex: starting from the raw parallel text, on which an SMT system is trained, to producing translations for an input sentence takes many steps and requires many components. I collectively refer to these steps as the *system*. Whereas the three parts of the proposed approach introduced in Section 1.2 are described in subsequent Chapters 3, 4, and 5 in great detail, I dedicate this section to the discussion of their use in the broader system.

The overall system architecture is illustrated in Figure 1.1. Components shown in colour correspond to the three parts of the approach presented in the subsequent chapters: components in red correspond to semantic data modelling (Chapter 3), components in green correspond to rule extraction (Chapter 4), and components in blue correspond to decoding (Chapter 5). The same system implementation is used for both translation and realization tasks by providing it with different data. For translation, the system is trained on parallel translation data; for realization, the system is trained on monolingual data. Additionally, the system uses monolingual data for language modelling and evaluation data for evaluation.

Although I do not discuss the implementation details of individual components and algorithms in this section, implementing all parts of the proposed approach was a major software engineering undertaking. It could not be accomplished without building on the work of others, in the shape of software packages, libraries, tools, and toolkits. I reference

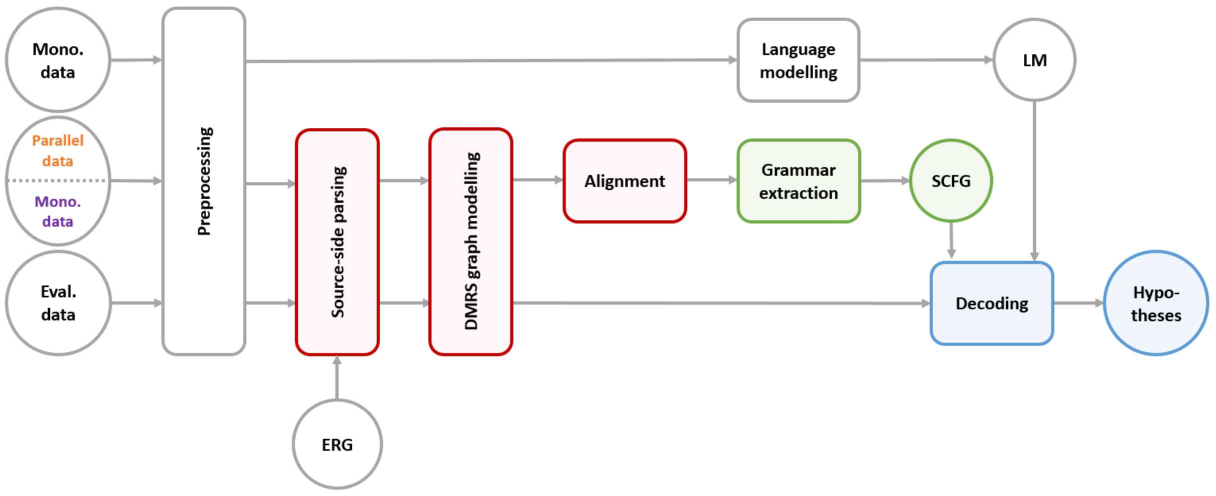


Figure 1.1: The overall architecture of the system proposed in this thesis. The individual system components are shown in rectangles, while the data they consume or produce is shown in circles. The colours group system components according to the chapters in which they are presented (red, Chapter 3; green, Chapter 4; blue, Chapter 5).

them throughout the thesis as appropriate. The bulk of the codebase implementing the proposed approach was written in the Python programming language, and is open-sourced (see Section 1.6).

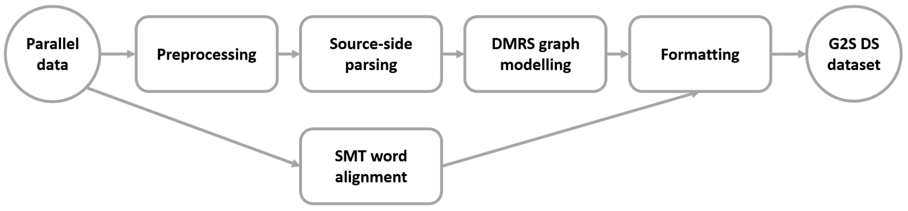
In the remainder of this section I give an overview of the system in greater detail (Section 1.5.1) and describe how the system is parallelized in order to allow large-scale processing (Section 1.5.2).

1.5.1 System overview

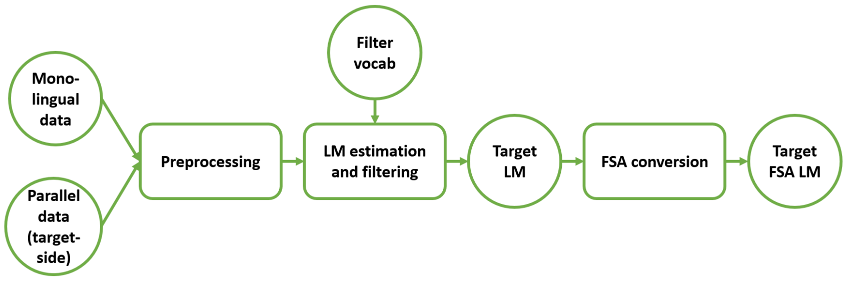
In order to get from raw parallel and monolingual data to a working statistical machine translation system a long series of steps needs to be completed. In Figure 1.1 I gave a general overview of the system. In this section I dissect the system into individual parts and describe them in greater detail. The system overview is aided by the diagrams shown in Figure 1.2. Each diagram corresponds to a different part of the system and, as before, rectangles represent system components while the data they produce or consume is shown in circles.

The parallel data preprocessing pipeline is illustrated in Figure 1.2a. In the first step, the preprocessing component, source and target sentences are cleaned and Unicode normalized. In order to reduce noise in the data, sentence pairs are filtered based on word fertility and detected language mismatch.[4] The sentences are tokenized, and in the case of German, truecased. After preprocessing, the source-side sentences are parsed with the English Resource Grammar and the resulting MRS representations are transformed into DMRS graphs. The DMRS graphs are modified and augmented as described in Section 3.2 and nodes and edge labels, and target-side tokens are id-mapped. SMT word alignment (see Section 3.3.3) is conducted for the translation task training data. The final step in

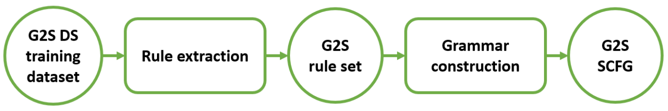
[4]Filtering based on word fertility refers to removal of sentence pairs with substantially different number of tokens; language detection filters a sentence pair if either of the sides deviates from expected language pairs.



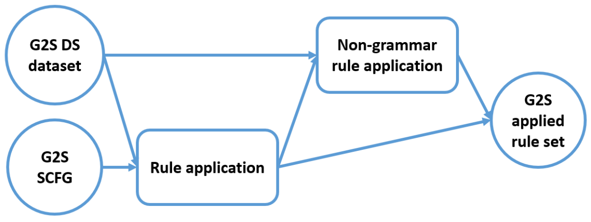
(a) Preprocessing pipeline.



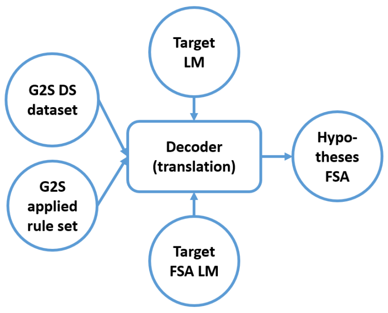
(b) Language modelling pipeline.



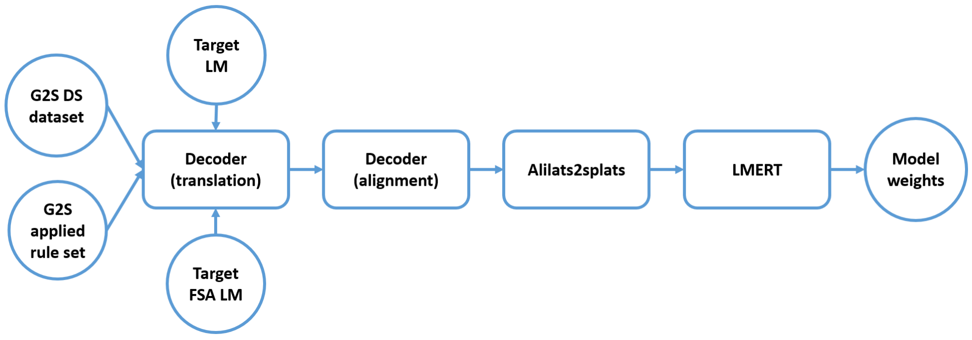
(c) SCFG extraction pipeline.



(d) Rule application pipeline.



(e) Translation mode decoding pipeline.



(f) Alignment mode decoding pipeline.

Figure 1.2: System overview diagrams. Each diagram corresponds to a different part of the system: (a) illustrates the preprocessing pipeline used by other parts of the system; (b) and (c) illustrate the training parts of the system; (d), (e), and (f) illustrate the decoding parts of the system. The individual system components are shown in rectangles, while the data they consume or produce is shown in circles.

the preprocessing pipeline is formatting, which combines all of the above information into a convenient XML format, which we will refer to as the DS format. The resulting graph-to-string dataset in DS format is ready to be used by rule extraction, rule application, or the decoder.

The language model estimation pipeline is illustrated in Figure 1.2b. Initially, target-language monolingual data and target side of the parallel data are preprocessed using the same preprocessing component as in Figure 1.2a. The cleaned, filtered, and tokenized sentences are used to estimate and filter an n-gram language model. Language model filtering to the parallel data target-side vocabulary reduces the n-gram language model size significantly. If the language model is to be used for local pruning (see Section 5.5.2), it is additionally converted into a Finite State Acceptor.

The synchronous context-free grammar extraction pipeline is illustrated in Figure 1.2c. The input to rule extraction (see Section 4.2) are training examples in the DS format, preprocessed using the preprocessing pipeline. The resulting set of graph-to-string rules is aggregated and rule features are computed to form a graph-to-string SCFG (see Section 4.3).

The rule application pipeline is illustrated in Figure 1.2d. The input is a tuning or testing dataset in the DS format, preprocessed using the preprocessing pipeline.[5] Given the input DMRS graphs and the graph-to-string SCFG, rule application (see Section 5.3) creates a set of applied rules and their corresponding graph coverages. Non-grammar rules (see Section 6.1) are added to the applied rule set in order to improve system robustness.

The decoding pipeline in translation mode is illustrated in Figure 1.2e. The decoder (see Section 5.4) represents each graph coverage as a Finite State Acceptor and from them recursively creates a hypothesis space FSA. The hypothesis space FSA is composed with an n-gram language model to get the final output of the decoder in translation mode.

The tuning pipeline for the log-linear model is illustrated in Figure 1.2f. The first part of the pipeline is identical to the translation mode decoding pipeline. After decoding in translation mode, the decoder is run in alignment mode with identical inputs, but constrained to the translation mode output (see Section 5.6.1). The resulting alignment FSTs are transformed into sparse FSTs, which encode rule feature contributions that are required by the Lattice Minimum Error Rate Training (LMERT) to optimize log-linear model parameters. The output of the tuning pipeline are updated log-linear model parameters, which can be used in the next tuning iteration.

1.5.2 Parallelization

Large-scale processing is an important aspect of the approach proposed in this thesis. It means that the system needs to be capable of extracting rules from millions of training examples and decoding real-world inputs using the resulting grammars. In addition to an efficient implementation, machine translation systems benefit from *parallelization.* Many processing tasks in machine translation in general, and in our system specifically, are *embarrassingly parallel.* That is, there is little to no effort required to decompose those tasks into parallel ones. This is the case for machine translation because the processing unit is a single sentence. Namely, decoding of each sentence is performed independently of other sentences.

[5]Only the source side of the tuning or testing dataset is preprocessed and no filtering is conducted.

In some machine translation subtasks, the information resulting from processing a single sentence needs to be aggregated in order to produce the desired output. An example of such a task is constructing the SCFG grammar from rule sets independently extracted from training examples. This and similar tasks (for example, Brants et al. (2007), Dyer et al. (2008), and Pino et al. (2012)) lend themselves well to a MapReduce programming model (Dean and Ghemawat, 2004). Apache Hadoop is a popular open-source distributed computing framework implementing the MapReduce programming model. An alternative to Hadoop is Apache Spark.[6] Compared to Hadoop, however, Spark enables a more complex dataflow and a wider range of operations in addition to map and reduce. Instead of storing all intermediate results on disk, it operates on data in-memory, which often makes it more efficient (Zaharia et al., 2010). I consequently use Apache Spark for applicable tasks in the proposed system.

In order to allow for large-scale processing with the proposed approach, several of the pipelines described in Section 1.5.1 are parallelized. Decoding in translation and tuning pipelines (Figures 1.2e and 1.2f) is parallelized by distributing sentence ranges as jobs on a grid computing cluster of machines.[7] The preprocessing pipeline including semantic data modelling (see Figure 1.2a) and SCFG extraction pipeline (see Figure 1.2c) are parallelized using Apache Spark. Apache Spark implementation of the rule application pipeline (see Figure 1.2d) has a particularly complex computational model since it parallelizes not only across sentences but also across SCFG rules. Grid computing cluster scripts and Apache Spark pipeline implementations are distributed alongside the open-source release of the codebase.

1.6 Contributions

In this section I collect and summarize the major thesis contributions discussed in the introduction and described in great detail in the subsequent chapters. The major contributions of this thesis are:

1. An architecture which combines the knowledge encoded in a deep hand-built grammar with a statistical machine translation approach. The approach consists of three parts:
    (a) Algorithms and approaches for processing of DMRS graphs and alignment of the graphs to source tokens, described in Chapter 3. In addition to the two tasks considered in this thesis, the algorithms and approaches are useful for a range of applications taking advantage of DMRS graphs, as discussed by Copestake et al. (2016).
    (b) A graph-to-string synchronous context-free grammar formulation and algorithms for extracting it from a parallel corpus, described in Chapter 4.
    (c) A rule application algorithm for applying the SCFG to graphs in decoding and an adaptation of the decoder by Iglesias et al. (2009) for decoding graphs into strings utilizing the mechanism of graph coverage, described in Chapter 5.

[6]http://spark.apache.org/
[7]Managed by Sun Grid Engine (SGE)

2. A demonstration that the approach is useful in practice:
   (a) as a large-scale machine translation system, demonstrated by applying the approach to a standard SMT English-German machine translation task, reported in Chapter 6.
   (b) as a robust statistical approach to realization of (D)MRS representations, filling in the gap where no robust realization alternative currently exists, reported in Chapter 7.

1.7 Publications

1. Matic Horvat and William Byrne. A Graph-Based Approach to String Regeneration. In *Proceedings of the Student Research Workshop at EACL*, pages 85–95, Gothenburg, Sweden, 2014
2. Matic Horvat, Ann Copestake, and William Byrne. Hierarchical Statistical Semantic Realization for Minimal Recursion Semantics. In *Proceedings of IWCS*, pages 107–117, 2015
3. Ann Copestake, Guy Emerson, Michael Wayne Goodman, Matic Horvat, Alexander Kuhnle, and Ewa Muszyńska. Resources for building applications with Dependency Minimal Recursion Semantics. In *Proceedings of LREC*, pages 1240–1247, 2016
4. Eunsol Choi, Matic Horvat, Jonathan May, Kevin Knight, and Daniel Marcu. Extracting Structured Scholarly Information from the Machine Translation Literature. In *Proceedings of LREC*, pages 421–425, 2016

Chapter 2

Related Work

The approach presented in this thesis builds on some of the machinery of syntax-based statistical machine translation (as discussed in Chapter 1). A notable influence on my approach is hierarchical phrase-based translation (Hiero) introduced by Chiang (2005, 2007). I describe hierarchical phrase-based translation in more detail in subsequent chapters: Hiero rule extraction is described in Section 4.1, while Hiero decoding is described in Section 5.1. In this chapter, however, I describe other related machine translation and realization approaches.

A common motivation for syntax-based SMT is to improve on phrase-based SMT approaches in terms of reordering and long-distance sentence dependencies. Reordering between source and target languages, particularly between syntactically diverging language pairs (e.g., SVO versus SOV in Japanese-English translation), is a weakness of phrase-based translation systems. Syntax-based SMT approaches address these problems by modelling the source-side syntax (Liu et al., 2006; Huang et al., 2006; Mi et al., 2008b), the target-side syntax (Yamada and Knight, 2001; Galley et al., 2006; Marcu et al., 2006), or both (Cowan et al., 2006; Zhang et al., 2008; Liu et al., 2009). I review them in Section 2.1.

Following the success of syntax-based SMT approaches, various approaches integrating deeper linguistic knowledge into SMT have been proposed. Several forms of semantics have been integrated into SMT systems. For instance, lexical semantics were integrated in phrase-based (Carpuat et al., 2006) and hierarchical phrase-based (Chan et al., 2007) translation. Semantic role labelling and predicate-argument structure have been used to enhance phrase-based (Xiong et al., 2012), hierarchical phrase-based (Gao and Vogel, 2011; Li et al., 2013), tree-to-string (Liu and Gildea, 2008, 2010; Wu et al., 2010b; Aziz et al., 2011), and string-to-tree (Bazrafshan and Gildea, 2013) translation. I review the semantics-based SMT approaches in Section 2.2.

Transfer-based machine translation (TBMT) approaches also frequently use semantics, although in the form of meaning representations. In TBMT the knowledge is encoded via a combination of hand-crafted and automatically acquired parsing, transfer, and realization rules. In Section 2.3 I discuss deep-syntax approaches to translation which follow the transfer-based MT paradigm, but introduce a statistical transfer component. I review the transfer-based MT approaches using Minimal Recursion Semantics in Section 2.4.

Jones et al. (2012) demonstrated that graph grammars are a promising approach for semantics-based machine translation. Consequently, I review graph grammar applications in Section 2.5.

Finally, in Section 2.6, I review notable statistical natural language generation approaches starting with the seminal work of Knight and Hatzivassiloglou (1995) and Langkilde and Knight (1998). I particularly focus on related approaches to generation that use statistical machine translation methodology.

2.1 Syntax-based SMT

In general, syntax-based SMT approaches aim to address deficiencies of phrase-based SMT systems, including reordering, particularly between syntactically diverging language pairs (for example, SVO vs. SOV). The syntax-based SMT approaches relevant to the work presented in this thesis and discussed here are approaches which directly model translation of or into syntactic structures (such as constituent or dependency trees), instead of using syntax as an additional feature or model in an otherwise non-syntactic system. There are several ways of categorizing the wide variety of approaches to syntax-based SMT.[1] For instance, based on the input, we can recognize string-based and tree-based approaches. The former accept strings as input and simultaneously parse and translate them. One of the earliest approaches to syntax-based translation is the string-based approach by Wu (1997), which simultaneously models source and target sentences. Other examples of string-based approaches include work by Chiang (2005, 2007) (further discussed in Sections 4.1 and 5.1), Galley et al. (2006) and Shen et al. (2008), discussed further below. Tree-based approaches instead accept trees as input and translate them either into target strings (Liu et al., 2006; Huang et al., 2006; Xiong et al., 2007; Mi et al., 2008b) or trees (Quirk et al., 2005; Ding and Palmer, 2005; Zhang et al., 2008).

Approaches to syntax-based SMT differ based on the formalism they use. Two classes of formalisms used most often are synchronous grammars and tree transducers. Synchronous grammars are a generalization of grammars, which generate a pair of strings or trees simultaneously. Synchronous grammars include inversion transduction grammar (ITG, used by Wu (1997)) and synchronous context-free grammar (SCFG, used by Chiang (2005, 2007)). The latter, for instance, is a generalization of context-free grammars, generating a pair of strings instead of a single string. Many approaches, however, trade computational efficiency for more expressive formalisms. Two popular formalisms are synchronous tree-substitution grammars (STSG; e.g., used by Eisner, 2003; Zhang et al., 2008; Chiang, 2010; Liu et al., 2009) and synchronous tree-adjoining grammars (STAG; e.g., used by DeNeefe and Knight, 2009; Liu et al., 2011), which are a generalization of STSG with tree adjunction. Shieber (2007) argues that STAG formalism is particularly well-suited to machine translation. An informal review of the SCFG, STSG, and STAG formalisms is given in Chiang and Knight (2006). Tree transducers are a generalization of Finite State Transducers. Graehl and Knight (2004) discuss various forms of tree transducers and how to train them. In particular, tree-to-string transducers (referred to as xRS) are often used in syntax-based translation (Yamada and Knight, 2001; Eisner, 2003; Marcu et al., 2006; Galley et al., 2006; Huang et al., 2006). A framework for unifying tree-transducer and synchronous grammar formalisms was proposed by Shieber (2004, 2006).

Another way of categorizing the syntax-based approaches is based on whether they are linguistically motivated or not, and if they are, which type of trees they use. Ap-

[1]This categorization is partially borrowed from Wu et al. (2010b).

proaches by Wu (1997) and Chiang (2005, 2007) are examples of formally-syntactic but not linguistically-motivated approaches. Most other work in syntax-based SMT is linguistically motivated, using constituent trees (Marcu et al., 2006; Galley et al., 2006; Liu et al., 2006; Huang et al., 2006; Mi et al., 2008b; Zhang et al., 2008; Liu et al., 2009) or dependency trees (Lin, 2004; Quirk et al., 2005; Ding and Palmer, 2005; Xiong et al., 2007; Shen et al., 2008, 2010; Tu et al., 2010). Below, I give more detail on linguistically-motivated approaches to syntax-based SMT, organised based on the type of tree used (constituent versus dependency) and directionality (string-to-tree, tree-to-string, tree-to-tree), while the formalism used is mentioned when appropriate.

2.1.1 String-to-tree translation

One of the earliest approaches to syntax-based translation is the approach by Yamada and Knight (2001). They describe a translation model that transforms parse trees into strings. The approach treats translation as a channel of operations in which child nodes in the source tree are reordered, new words are inserted, and finally the leaf words are translated. Since the SMT system is based on the noisy-channel model, the decoder using the above translation model works in the reverse direction of the channel. The decoding algorithm, presented in Yamada and Knight (2002), therefore builds a target tree based on an input string (similarly to how a parser builds a parse tree), using the tree-to-string transducer (xRS) formalism. Translation approaches of this type are often referred to as string-to-tree (or S2T) translation.

The first syntax-based translation system that outperformed a phrase-based baseline system in a large scale evaluation was presented by Marcu et al. (2006). Dubbed SPMT, the approach translates input strings using translation rules with syntactified target language phrases to derive target language parse trees. A more expressive approach compared to SPMT was presented by Galley et al. (2006). They extend the GHKM string-to-tree rule extraction algorithm introduced by Galley et al. (2004) to extract larger (combined) rules. Both SPMT and the approach by Galley et al. (2006) use the xRS formalism. Although the improvements to the GHKM algorithm result in improved performance compared to the baseline GHKM, the approach achieves significantly worse performance on a large-scale translation task compared to the SPMT system. DeNeefe et al. (2007) improve phrasal coverage of a syntax-based translation system compared to a phrase-based system by combining the minimal and composed GHKM rules of Galley et al. (2006) with the SPMT model 1 of Marcu et al. (2006), along with tree binarization of Wang et al. (2007). DeNeefe and Knight (2009) approach S2T translation by using synchronous tree adjoining grammar formalism.

2.1.2 Tree-to-string translation

An alternative to string-to-tree translation is tree-to-string translation (T2S), which models the syntax of the source language instead of the target language. Tree-to-string translation is performed in two steps: (1) parsing of the input string to obtain a syntax tree, and (2) decoding the syntax tree with T2S transformation rules. The task of T2S decoder is therefore to find the best target string instead of finding the best target tree. Tree-to-string decoders are often significantly faster than their S2T counterparts (Huang et al., 2006). Liu et al. (2006) apply tree-to-string alignment templates to transform a constituent tree into a target string. T2S alignment templates describe alignment between a source parse tree and a target string and are used to produce candidate translations in bottom-up beam search. In contrast, Huang et al. (2006) extend tree-to-string transducers to recursively convert input trees to strings. Based on lessons from phrase-based decoding, Huang and Mi (2010) significantly improve T2S translation efficiency by developing an incremental dynamic programming algorithm, making T2S translation 30 times faster than the Moses phrase-based translation system (Koehn et al., 2007). Liu et al. (2011) introduce synchronous tree adjoining grammar to T2S translation and cast decoding as a tree parsing problem, while Xiao et al. (2014) use tree-to-string rules obtained via GHKM algorithm to complement hierarchical phrase-based rules.

Neubig and Duh (2014) investigated why T2S translation systems are reported to achieve state-of-the-art performance in some experiments, but lag behind other approaches in others. They argue that T2S have the potential to achieve high performance, but are sensitive to peripheral elements of T2S translation. By evaluating several systems on Japanese-English and English-Japanese translation tasks, they found that parser accuracy, alignment quality, and search algorithm all have significant effects on the final system performance. In addition to their findings on the effect of parse quality on T2S translation, Neubig and Duh (2014) found that using a forest of parse trees instead of 1-best parse improved the translation performance further. Zhang and Chiang (2012) found that using a forest improves on T2S translation because it allows the translation system to override the syntactic constraints imposed by a single tree.

The extension from a single parse tree to a forest of parse trees is referred to as forest-to-string translation (F2S). Mi et al. (2008b) introduced a translation system that translates a packed forest containing exponentially many parses (namely, many more than a k-best list of parses, which contains too few variations). As in T2S translation, F2S decoding proceeds in two steps: (1) the input is parsed into a packed forest (e.g., a hypergraph), which is subsequently (2) converted into a translation forest via pattern-matching techniques similar to the ones used in T2S translation. Additionally, forest pruning is used prior to the conversion to reduce the size of the packed forest. In complementary work, Mi et al. (2008a) introduce rule extraction from a packed forest representation with the aim of alleviating the consequences of parsing errors on extracted rule quality. They show that forest-based rule extraction benefits the performance of tree-to-string and string-to-tree translation approaches, but achieves the best performance with the forest-to-string decoder of Mi et al. (2008b) (referred to as the forest-forest system).

2.1.3 Tree-to-tree translation

In tree-to-tree (T2T) translation, source parse trees are mapped to target parse trees. One of the earliest approaches to T2T translation is by Cowan et al. (2006) in which source parse trees are broken down into clauses and translated independently in sequence. Eisner (2003) presented algorithms for tree-to-tree alignment, training, and decoding under the STSG formalism. Approaches by Zhang et al. (2008) and Chiang (2010) use STSG for tree-to-tree translation. Liu et al. (2009) pose that tree-to-tree translation approaches thus far have been underperforming due to parse errors affecting 1-best T2T translation and poorer rule coverage in comparison to S2T and T2S translation due to the lack of linguistically-unmotivated mappings. They propose to use packed forests in a forest-based tree-to-tree translation model using STSG learned from aligned forest pairs. The decoder decomposes trees in the source forest while building target forest in parallel. Using this approach they achieve significant improvement over using 1-best trees (3.6 BLEU points).

2.1.4 Dependency tree translation

Instead of using constituent trees, Lin (2004) describe a model which uses dependency trees on the source side and their mapping on the target side (i.e., not actual dependency parses of the target side). The model translates paths in the source dependency tree into mapped dependency tree fragments on the target side. Similarly to Lin (2004), treelet translation approach by Quirk et al. (2005) projects the parsed source dependency structure onto the target sentence in order to extract treelet phrase pairs (treelet is a connected subgraph of the dependency tree). Using a tree-based reordering model, the decoder searches for the best combination and ordering of treelet translation pairs in order to translate the input dependency graph. Instead of projecting the source dependencies to the target side, Ding and Palmer (2005) uses both source and target dependency trees to estimate a translation model based on synchronous dependency insertion grammar (SDIG) and create target-side dependency structures using a non-isomorphic tree-to-tree transducer.

In contrast to previous dependency translation approaches, dependency treelet string correspondence (DTSC) approach by Xiong et al. (2007) extends the treelet concept from Quirk et al. (2005) to map source dependency structures to target strings directly. Since the previous approaches lack reordering information specified in translation rules and make up for it with heuristics (Lin, 2004; Xiong et al., 2007) or separate ordering models (Quirk et al., 2005; Ding and Palmer, 2005), Xie et al. (2011) introduce head-dependents rules, composed of a head and all its dependents on the source side, which specify the ordering information directly.

Li et al. (2015) follow the rule extraction approach by Chiang (2007) to estimate a synchronous edge replacement grammar (a graph grammar, further discussed in Section 2.5) on pairs of edge-labelled dependency graphs and target strings. The converted dependency trees are then translated using the CYK parsing algorithm. Li et al. (2016) present a graph-based translation model that translates a graph into a string by segmenting the graph into pieces, rather than extracting a recursive grammar. The source-side graphs are augmented dependency trees with bigram edges representing the source word order.

Analogous to the difference between T2S and S2T translation, dependency trees can be built on the target side. SDIG approach by Ding and Palmer (2005), introduced above, is one such example. Shen et al. (2008) (and later Shen et al. (2010)) introduce string-to-dependency translation, which builds dependency structures on the target side in a chart parsing approach modelled after Chiang (2007). Instead of an n-gram language model, the approach uses a dependency language model during decoding. With similar motivation as F2S rule extraction by Mi et al. (2008a), Tu et al. (2010) formulate a dependency forest for string-to-dependency rule extraction and dependency language model estimation, which improve the string-to-dependency translation approach by Shen et al. (2008).

2.2 Semantics-based SMT

Following syntax-based approaches to SMT, a number of approaches integrating various forms of semantics into SMT have been proposed. Dependency graph-based approaches discussed in Section 2.1.4 are sometimes considered as shallow semantics. Lexical semantics using word sense disambiguation (WSD) were one of the earliest form of semantics integrated into SMT (Carpuat et al., 2006; Chan et al., 2007; Carpuat and Wu, 2007a,b). Particularly relevant to the approach presented in this thesis are approaches to SMT using semantic role labelling as well as approaches using semantic representations of the entire sentence. I discuss the translation approaches using semantic role labelling (SRL) and predicate-argument structures (PAS) in Section 2.2.1. Finally, in Section 2.2.2, I describe approaches to translation using Abstract Meaning Representation (AMR).

(Dependency) Minimal Recursion Semantics used in this thesis (introduced in Section 3.1) shares similarities with the various forms of semantics discussed in this section. With regards to lexical semantics, MRS representations disambiguate word senses to the extent to which the senses affect the syntax, whereas WSD approaches disambiguate them further. Semantic role labelling is the task of finding semantic roles for predicates in a sentence, such as `Agent`, `Patient` etc. PropBank (Palmer et al., 2005) and FrameNet (Baker et al., 1998) are two types of commonly used semantic roles. Semantic role labelling is closely related to predicate-argument structures: whereas predicate-argument structure denotes which predicates and arguments share a relationship, semantic roles denote the type of relationship between them. Despite the similar naming scheme to PropBank (Palmer et al., 2005), the notion of roles as used in SRL is somewhat different from the arguments as used in MRS. On the other hand, an MRS can be written as a predicate-argument structure. A discussion on the relationship between SRL and MRS and on the potential addition of semantic roles to MRS can be found in Copestake (2009). In comparison to MRS, Abstract Meaning Representations are derived non-compositionally and are consequently further abstracted from the sentence syntax than MRS (see Bender et al. (2015) for a detailed discussion).

2.2.1 Translation with SRL and PAS

Approaches for integrating semantic role labelling and predicate-argument structure information into statistical machine translation are motivated by the studies of (1) Fung et al. (2006), who investigated cross-lingual semantic verb frame argument mappings between Chinese and English and found that 84% of semantic role mappings remained consistent; and (2) Wu and Fung (2009b), who investigated potential contributions of semantic roles to SMT.

Semantic role label information has most often been used to enhance an existing translation system, as opposed to translating these structures directly. One of the earliest approaches utilizing SRL, introduced by Wu and Fung (2009a), proposes a two-pass architecture in which a phrase-based SMT system first creates a hypothesis, followed by reordering of the constituent phrases in order to maximize cross-lingual semantic role label match between the source and target sentences. However, semantic role label information is more commonly integrated into an existing translation system in the form of a model used as a feature. Xiong et al. (2012) introduce two discriminative models of source-side predicate argument structure into a phrase-based translation system: (1)

predicate translation model uses lexical and semantic contexts to translate verbal predicates; (2) argument reordering model uses various source and target-side features to model verbal argument reordering between source and target side.

Liu and Gildea (2010) incorporate semantic role information into a tree-to-string transducer via features modelling reordering and deletion of semantic roles. The target-side semantic roles are created by projecting source semantic role labels to the target side via word alignments. Li et al. (2013) present an approach to integrating SRL information into hierarchical phrase-based translation system. In order to enable the use of semantic role labels, they constrain the Hiero system with hard syntactic constraints. Similarly to Liu and Gildea (2010), they project the source semantic role labels to the target side and model the predicate-argument reordering between them. They integrate the model into the constrained Hiero as a feature. Li et al. (2014) present a unified framework for integrating soft linguistic (both syntactic and semantic) reordering constraints into Hiero. Semantic constraints are based on the predicate-argument structures. The constraints are incorporated into Hiero as a feature computed during decoding.

In addition to integrating SRL as a feature, SRL can be used to constrain or augment rule extraction in syntax-based systems. Liu and Gildea (2008) augment the source-side syntactic trees with semantic roles and use them in a tree-to-string translation system. Similarly, semantic role labels are added to source-side syntactic trees in the T2S translation approach by Aziz et al. (2011). Wu et al. (2010b) propose to use deep syntactic information based on an HPSG parse for tree-to-string translation. Their composed translation rules are extracted based on the predicate-argument structure derived from the HPSG parser (T2S rule extraction is detailed in Wu et al. (2010a)). Gao and Vogel (2011) integrate SRL into Hiero by extracting SRL-aware SCFG rules in addition to existing Hiero rules. SRL-aware rules incorporate target-side SRL information in their non-terminals and take precedence over regular Hiero rules during decoding. Similarly to the approach by Gao and Vogel (2011), Bazrafshan and Gildea (2013) extend a string-to-tree SMT approach with semantic role label information by (1) augmenting target-side syntax trees with semantic role label information before extracting rules with a modified GHKM algorithm; and (2) extracting rules with the argument structure of a single predicate, in addition to regular S2T rules. They extend their work in Bazrafshan and Gildea (2014) by extracting the predicate-argument structure rules for both source and target sides, instead of just target sides. Additionally, they present an SRL language model, defining a probability distribution over sequences of semantic roles on the target side.

Rather than using SRL for modelling source-target translation and reordering, Haque et al. (2011) use semantic role labels as source contextual features with the aim of improving source sense disambiguation in a phrase-based system. Furthermore, SRL has recently been used to encode information about lexical semantic affinities between predicates and their arguments. Tang et al. (2016) use selectional preferences to put semantic restrictions on verb arguments, based on semantic role labels. Nadejde et al. (2016) introduce a selectional preference feature, modelling selectional preference of verbs on the target side for their arguments (similar to work by Tang et al. (2016)) and nouns for their prepositional arguments. Whereas Tang et al. (2016) integrate their model into a phrase-based system, Nadejde et al. (2016) integrate the feature into a string-to-tree system.

Instead of incorporating SRL information into an existing translation system, Zhai et al. (2012) propose a standalone predicate-argument structure translation approach. The approach transforms source PAS into target PAS, which subsequently provides the ordering in which source elements are individually translated. The target PAS can be thought of as a skeleton translation. The source to target PAS transformation rules are obtained automatically from bilingual semantic role labelling. Even though the PAS-based translation approach can function independently, it achieves the best results when combined with an established translation system. Zhai et al. (2013) extend the PAS-based translation approach by identifying two types of PAS ambiguities, gap and role ambiguity, and proposing two methods to aid PAS disambiguation by using additional context.

2.2.2 AMR-based translation

An investigation on the suitability of AMRs as a minimally divergent transfer layer in machine translation was carried out by Xue et al. (2014), comparing English AMRs to Chinese and Czech AMRs. They concluded that using AMRs as a transfer layer would require large and complex entries in the translation dictionary and it may therefore be more appropriate to use abstract meaning representations just on the source or target side.

Tamchyna et al. (2015) present an approach to use source-side semantic information in the form of an AMR graph to rescore the outputs of a phrase-based translation system. Jones et al. (2012) present a semantics-based approach to machine translation using AMR (although the approach is flexible in terms of the meaning representation, the only requirement being that it is a directed acyclic graph). The approach uses two synchronous hyperedge replacement grammars (SHRGs): source SHRG is used to transform the source language string into a meaning representation graph, while target SHRG is used to transform the 1-best graph into the output string. The SHRG grammars are induced between surface forms and meaning representation with either of two grammar induction algorithms presented in the paper. Synchronous hyperedge replacement grammar is an instance of a graph grammar, which I review in Section 2.5.

Notably, the approach does not rely on graphs containing any alignment information. Consequently, authors introduce a string-graph alignment algorithm. Similarly, Pourdamghani et al. (2014) present a generative model and use EM to estimate alignments between AMR graphs and strings. In contrast, the DMRS string-graph alignment procedure (described in Section 3.3.3) relies on the source string alignment information associated with each DMRS node, obtained as a by-product of the compositional parsing procedure.

2.3 Deep-syntax translation

Lexical Functional Grammar (LFG) is a hand-built deep grammar and is to an extent similar to the English Resource Grammar used in this thesis. In particular, LFG f-structures approximate to basic predicate-argument-adjunct structures or dependency relations (Cahill and van Genabith, 2006). However, MRS representations are more directly related to the LFG Glue Language / s-structures, since, for instance, f-structures do not represent scope.

LFG f-structures were used in several transfer-based machine translation approaches. A TBMT approach commonly consists of three steps: (1) parsing of the source sentence into a source meaning representation, (2) transferring the meaning representation into a target meaning representation, and (3) realizing the target meaning representation into the target string. Each step of the transfer-based MT pipeline is either fully hand-crafted, fully statistical, or hybrid. Approaches with a statistical or hybrid transfer step are relevant to the approach described in this thesis.

Riezler and Maxwell III (2006) present a TBMT approach using the LFG f-structures with a statistical transfer component and hybrid approaches for parsing and realization from f-structures. Transfer rules, mapping between dependency feature structures, are acquired automatically from a parallel corpus. The transfer rules are used in a decoder modelled after Pharaoh (Koehn et al., 2003), a phrase-based translation decoder. Instead of LFG, Bojar and Hajič (2008) describe a deep-syntax translation approach with a statistical transfer component using the Functional Generative Description (FGD) Tectogrammatical layer. Their system is based on the STSG formalism (see syntax-based SMT approaches in Section 2.1).

Particularly relevant to my work is the f-structure TBMT approach by Graham et al. (2009) with a statistical transfer component. The decoder uses transfer rules automatically induced from a parallel corpus using a procedure described in Graham and Genabith (2009). Although the approach is unable to surpass a baseline Moses system, it demonstrates improved performance on subsets of data,[2] in particular translation of noun compounds (Graham, 2012).[3]

The approach by Graham et al. (2009) shares similarities with the one presented in this thesis. Namely, both approaches are statistical and use a similar source of monolingual knowledge (i.e., a hand-built deep grammar). Nevertheless, the approaches differ significantly in that Graham et al. (2009) use primarily a transfer-based approach to MT and methodology, whereas I use statistical MT approach and methodology.[4]

2.4 Machine Translation with MRS/DMRS

The translation approach described in this thesis is based on (Dependency) Minimal Recursion Semantics representations (introduced in Section 3.1). Minimal Recursion Semantics was introduced in the context of a transfer-based translation system (Copestake et al., 1995) and has since been used in several transfer-based and other approaches to machine translation. I discuss the most notable approaches in this section.

Oepen et al. (2004a) introduce the LOGON initiative, a large-scale effort for high-precision Norwegian machine translation. The Norwegian-English transfer-based translation system is comprised of: (1) source-side parsing with a Norwegian LFG grammar and projection

[2]Although certainly not a toy system, the evaluation in Graham (2012) is conducted on a smaller scale than in this thesis, using 360 thousand training examples compared to 4.25 million (see Chapters 6 and 7).

[3]A major problem for the system is generation from flawed representations. Consequently, it may benefit from using the approach proposed in this thesis as a robust realization system (provided it is adapted to be used with LFG f-structures). In Chapter 7 I demonstrate its potential as a realization component in a different transfer-based MT approach.

[4]In some respects, my approach could be seen as transfer-based MT approach with a joint transfer and realization component. However, that would diminish the prominence of *transfer* in transfer-based machine translation.

to MRS structures; (2) MRS-based transfer with rewriting rules; and (3) realization of transferred MRS to English sentences with the ERG. Each of the steps in the transfer pipeline can produce a number of outputs (e.g., a few hundred). Since the fan-out combinatorics are potentially prohibitive for exact exploration, n-best lists are considered after each step. The rule-based components in the translation system are used to produce grammatically and semantically coherent translations, whereas the ranking of competing hypothesis is seen as a probabilistic task. Under that premise, Oepen et al. (2007) use ranking models after every step of the transfer pipeline. While they use an existing parse selection model, they describe new models for ranking of transfer outputs and target realizations. Finally, they introduce a discriminative model for end-to-end reranking of the complete list of candidate translations.

Building on the transfer infrastructure of the LOGON project, Bond et al. (2005) present a fully open-source proof-of-concept Japanese-English MT system. Unlike in the work by Oepen et al. (2004a), however, the source language sentence is parsed using the HPSG-based JACY grammar (Siegel, 2000; Siegel et al., 2016), instead of using an LFG-based grammar. The resulting MRS representations are transferred using a small set of hand-constructed transfer rules and a small transfer lexicon. The English MRS representations are realized with the ERG. Nichols et al. (2007) extend the work of Bond et al. (2005) on Japanese-English translation by introducing several additions: (1) using reranking models of outputs at all steps as proposed by Oepen et al. (2007); (2) automatically obtain open-class transfer rules from a bilingual dictionary; and (3) combining the rule-based approach with an SMT system (Moses) to improve system robustness. Bond et al. (2011) describe further development efforts on the open-source Japanese-English translation and compare its performance to a phrase-based translation system. In addition to obtaining open-class transfer rules from bilingual dictionaries (as in Nichols et al. (2007)), they obtain them from parallel corpora using Moses-generated phrase tables (in total, the system uses 105,000 automatically obtained transfer rules). The method for rule construction from parallel corpora is described by Haugereid and Bond (2012).

Instead of a transfer-based approach, approaches by Wang et al. (2012), Simov et al. (2015), and Simov and Osenova (2016) all use parts of (R)MRS analysis to enhance a factor-based statistical machine translation approach for Bulgarian-English and/or English-Bulgarian translation.

2.5 Graph grammars

Hyperedge replacement grammars (HRG) are a generalization of context-free grammars to hypergraphs. Like synchronous context-free grammars, synchronous hyperedge replacement grammars used by Jones et al. (2012) are an extension of HRGs with source and target side for every rule. Drewes et al. (1997) give a comprehensive survey of HRGs, while Bauer and Rambow (2016) study some of their limitations. In addition to machine translation, SHRGs have also been used for parsing strings into meaning representation graphs (Chiang et al., 2013; Peng et al., 2015).

HRG and SHRG are examples of (context-free) graph grammars, which are more powerful and versatile than their string counterparts. However, the price of higher expressiveness is increased computational complexity (Drewes et al., 2015). Consequently, approaches in natural language processing that use graph grammars are usually evaluated on a small scale. For example, (1) Jones et al. (2012) evaluated their approach on an English-Chinese translation task using GeoQuery data (Tang and Mooney, 2001; Lu and Ng, 2011), with 880 parallel sentences, 280 of which were used for evaluation; (2) Jones et al. (2013) evaluate their approach to automatic extraction of HRGs on 5564 elementary semantic dependency graphs (Oepen and Lønning, 2006) taken from the LOGON portion of the Redwoods corpus (Oepen et al., 2004b) (like DMRS, elementary semantic dependency graphs are derived from Minimal Recursion Semantics representation); (3) Peng et al. (2015) evaluate their SHRG parsing approach on 2132 sentences with 3955 training and 2132 development sentences.

Despite the fact that the applications of graph grammars, like in examples above, often go significantly beyond toy example demonstrations, they are far removed from the large-scale evaluations commonly conducted in statistical machine translation and presented in this thesis (see Chapters 6 and 7). Nonetheless, graph grammars are part of an active research area on formal models of graph transformation whose major focus is improving efficiency of graph grammar algorithms. A recent seminar titled "Formal Models of Graph Transformation in Natural Language Processing" brought together researchers in graph transformation and researchers in natural language processing in order to "assess the state of the art, identify areas of common interest, and pave the way for future collaborations" (Drewes et al., 2015).

A linguistically-motivated graph grammar formalism is s-graph grammars. Koller and Kuhlmann (2011) introduced interpretable regular tree grammars (IRTGs) that generalize over existing formalisms, including synchronous context-free grammars and tree transducers. The IRTG formalism is extended by Koller (2015) to the synchronous case of mapping between strings and graph-based semantic representations. The new formalism is called s-graph grammar. The author shows its potential use for graph-based compositional semantic parsing (or as they refer to it, construction). Notably, Groschwitz et al. (2015) use s-graph grammars and present two algorithms for realization of strings from a graph meaning representation. They evaluate the approach on a small-scale task of 1562 AMR-sentence pairs.

Braune et al. (2014) contrasts three formalisms for capturing the relation between semantic graphs and English strings: (1) synchronous hyperedge replacement grammars, (2) DAG-to-tree transducer (D2T), and (3) cascade-to-tree transducer (Cascade). They manually constructed grammars to parse into and generate from 10,000 automatically constructed graphs. They found that Cascade needed fewest rules and D2T the most. All approaches performed well in terms of generation coverage (100%) and accuracy, but less so with parsing coverage (around 5%). They conclude that the number of required rules remains too high and that none of the formalisms elegantly captures the full complexity of the dataset, especially for the parsing task.

2.6 Statistical realization

In this section I review some notable statistical realization approaches, but focus in particular on the realization approaches that take advantage of machine translation machinery. A more extensive review of statistical approaches to realization can be found in Velldal (2007).

The seminal work that spurred statistical approaches in the natural language generation field was the Nitrogen system by Knight and Hatzivassiloglou (1995) and Langkilde and Knight (1998). The Nitrogen system operates in two stages, first generating all possible sentences from a given semantic representation, encoded in a word lattice, followed by selecting the best one using an n-gram language model. Langkilde (2000) improves its performance by replacing the word lattice with a more compact forest structure. Langkilde-Geary (2002) introduce Halogen, the successor to Nitrogen, which uses the same two-stage approach as Nitrogen but improves on it in several aspects. The input to the Nitrogen and Halogen system are Abstract Meaning Representations, a precursor to modern-day AMR (Banarescu et al., 2013).

In terms of methodology, the work presented in this thesis is perhaps closest to Cahill and van Genabith (2006) whose PCFG system for robust probabilistic realization is based on approximations to an LFG automatically extracted from a treebank. The approach by White (2004) performs realization from logical form inputs with a chart-based algorithm employing a grammar. In this respect it is similar to the established MRS realization approach which also employs chart-parsing with a grammar (Carroll et al., 1999).

Statistical machine translation approaches have previously been employed for tactical generation. Wong and Mooney (2007) propose three methods for tactical generation from formal meaning representations based on statistical machine translation on a limited domain (RoboCup and GEOQUERY datasets): (1) using a phrase-based SMT system on linearised parse trees, (2) inverting an SMT-based semantic parser based on SCFG, and (3) a hybrid of the first two approaches. A related approach by Lu and Ng (2011) adapts the SCFG formalism for realization from logical forms (typed lambda calculus). White (2011) investigates an approach which has some similarities to the one proposed in this thesis, using MT-style glue rules for robustness in conjunction with a realizer based on CCG. However, his approach is directed at patching up failed realizations. In the approach proposed by Flanigan et al. (2016), the AMR graph is transformed into a spanning tree and subsequently decoded into a string using a tree-to-string transducer and language model. The approach is trained on the pairs of treebanked AMR graphs and corresponding strings. A competing approach by Pourdamghani et al. (2016) instead learns to linearise AMR graphs into strings and subsequently uses a standard phrase-based machine translation system for realization. The approach improves significantly on the performance by Flanigan et al. (2016).

Due to the significant differences between meaning representations used (in terms of their structure and the degree to which they pre-determine their natural language realizations), the realization evaluation methodology is not as standardised. The first surface realization shared task (Belz et al., 2011) was an attempt to standardise the surface realization evaluation in order to accelerate the progress in surface realization approaches. Unfortunately, the task did not succeed in setting a standardised evaluation methodology (Belz et al., 2012).

Chapter 3

Semantic data modelling

In the first part of this chapter I introduce the Dependency Minimal Recursion Semantics (DMRS), a semantic representation of a sentence that I use throughout the thesis. DMRS is a directed acyclic graph with nodes corresponding to predicates and edges representing the relationships between them. DMRS was introduced by Copestake (2009) as an easy-to-read and easy-to-use alternative to Minimal Recursion Semantics (MRS) representation (Copestake et al., 1995, 2005). DMRS and MRS representations are interconvertible between each other without any loss of information. MRS representations are grounded in formal logic, such as predicate calculus. The standard method of obtaining an MRS or DMRS representation of a sentence is via parsing using a hand-crafted grammar. The grammar used in this thesis is the broad-coverage English Resource Grammar (ERG; Flickinger, 2000). In Section 3.1 I provide a more detailed introduction to MRS, DMRS, and related concepts.

DMRS was designed to include all semantically related information that can be derived from syntax and morphology. However, not all information present in the representation is useful for its applications. Moreover, some information may harm system performance. Consequently, MRSs are often transformed for use in applications in an application-specific way (for example, Herbelot and Copestake (2006); Bond et al. (2011); Packard (2014); Yin et al. (2014)). In the second part of this chapter I describe three methods for transforming DMRS graphs in order to make them more suitable for use with the statistical machine translation approach and translation and realization tasks considered in this thesis. The methods include node and edge labelling, grammar predicate filtering, and removal of cycles in the underlying undirected graph. The latter two methods are, by design, lossy and result in DMRS graphs which cannot be converted back into MRS form. I describe the methods in detail in Section 3.2. Although the methods were designed to make the DMRS graphs more suitable for the statistical machine translation approach and translation and realization tasks, they have potential to be used in other application-specific modelling of DMRS graphs. This is substantiated by their inclusion in the pydmrs library released alongside the publication by Copestake et al. (2016).

The statistical machine translation approach proposed in this thesis is trained on a parallel corpus of examples. Specifically, a synchronous context-free grammar (described in Chapter 4) is extracted from training examples consisting of a source-side DMRS graph (after modelling), a target-side tokenized sentence, and the alignment between the two. In the final part of this chapter I describe the methods for creating an alignment between the DMRS graph nodes and sentence tokens. The alignment between the two is based on the alignment information provided as a result of parsing, but needs to be transformed into a form more suitable for the statistical machine translation approach. In particular, the original alignment is based on a different tokenization scheme and omits semantically empty words. Furthermore, the alignment needs to be extended to the target-side language for the translation task, which is accomplished using statistical word alignment. I describe both source-side and target-side alignment in detail in Section 3.3.

3.1 Dependency Minimal Recursion Semantics

Minimal Recursion Semantics (MRS) is a framework for computational semantics introduced by Copestake et al. (1995) and formally described in Copestake et al. (2005). MRS was designed to be a tractable representation for large-scale parsing and realization, while not sacrificing expressiveness. It provides flat semantic representations that enable underspecification and can be integrated with grammatical representation in a number of frameworks. **Dependency Minimal Recursion Semantics** (DMRS), introduced by Copestake (2009), is an alternative representation inter-convertible with MRS. It has minimal redundancy in its structure and was developed for the purpose of readability and ease of use for both humans and computational applications. While MRS and DMRS have been used in a wide variety of grammars, I concentrate here on the output by the English Resource Grammar (ERG, Flickinger (2000)). The descriptions and examples in this section and throughout the thesis will use the 1214 version of ERG.

In the remainder of this section I describe MRS (Section 3.1.1) and DMRS (Section 3.1.2) in more detail using an example sentence. I conclude by briefly describing the English Resource Grammar and parsing in Section 3.1.3.

3.1.1 Minimal Recursion Semantics

MRS is a meta-language for describing semantic structures in some underlying object language: the object language usually discussed and used here is predicate calculus with generalized quantifiers. To illustrate MRS, consider the sentence shown in (1) and the corresponding (simplified) MRS in (2):

(1) No generally accepted formal definition of algorithm exists yet.

(2) **LTOP**: `l2`,
INDEX: `l2`,
RELS: < `l4:_no(x, h7, _), l8:_general(e1, e2), l8:_accept(e2, _, x),`
`l8:_formal(e3, x), l8:_definition(x, y), l5:udefq(y, h6, _),`
`l3:_algorithm(y), l2:_exist(e4, x), l2:_yet(e5, e4),` >
HCONS: < `h7` $=_q$ `l8, h6` $=_q$ `l3` >

The main part of the MRS representation is the RELS list containing a bag of elementary predications (EPs). Each EP has an associated label: for example, `l8: _definition(x, y)` has label `l8`. Predicates corresponding directly to word lemmas are indicated with a leading underscore. The other class of predicates are grammar predicates (sometimes referred to as gpreds), which are used to convey additional information about the existing predicates. For example, `udefq` is a placeholder quantifier used in absence of other quantifiers. `udefq(y, h6, _)` in (2) is used for that purpose for the bare singular *algorithm*.

Local top (LTOP) is the topmost label in the MRS that is not the label of a quantifier. Index (INDEX) is an important feature in HPSG (as described in Pollard and Sag (1994)), which, at sentence level, points to the syntactic head of the sentence, which is nearly always the main verb of the sentence. In (2), both LTOP and INDEX point to the same label, `l2`, associated with `l2:_exist(e4, x)` and `l2:_yet(e5, e4)` elementary predicates.

Finally, HCONS is a set of constraints which relate labels to argument 'holes' in quantifiers and other scopal predicates. For example, `_no_q(x, h7, _)` is a scopal predicate whose hole is denoted as `h7`. The constraint `h7` $=_q$ `l8` therefore specifies a qeq relationship between `h7` and `l8`.

In order to explain qeq constraints and underspecification, let's examine (3), which shows the scoped readings of (1) in the underlying object language:

(3) a. udefq(y, _algorithm(y), _no(x, _general(e1, e2) ∧ _accept(e2, _, x) ∧ _formal(e3, x) ∧ _definition(x, y), _exist(e4, x) ∧ _yet(e5, e4)))

b. _no(x, udef(y, _algorithm(y), _general(e1, e2) ∧ _accept(e2, _, x) ∧ _formal(e3, x) ∧ _definition(x, y)) _exist(e4, x) ∧ _yet(e5, e4))

To show the relationship between these structures and MRS, first observe that through nesting of arguments each reading forms a tree, and that elements at each level of the tree are always combined with conjunctions. The root of the tree corresponds to the topmost quantifier. The MRS representation uses a list of predications instead of the explicit conjunction. An element can be added to each predication to express its position in the tree: this is the MRS label. Instead of expressing the relationship between the quantifier (or other scopal predicate) and its arguments by embedding, MRS uses a 'hole' argument to the quantifier and equates it to a label, as shown in (4):

```
(4) l5:udefq(y, h6, l4), h6=l3, l4:_no(x, h7, l2), h7=l8,
    l8:_general(e1, e2), l8:_accept(e2, _, x), l8:_formal(e3, x),
    l8:_definition(x, y), l3:_algorithm(y), l2:_exist(e4, x),
    l2:_yet(e5, e4)
```

Specifying an LTOP and putting the hole-label equalities into HCONS results in what is now formally an MRS. However, the MRS only corresponds to the first reading shown in (3). Scope underspecification in MRS is a generalization of the trees corresponding to the different scopes, maintaining the constraints between the elements via qeq constraints ($=_q$, equality modulo quantifier) between hole arguments and labels. Intuitively, a qeq constraint, $h =_q l$, enables one or more quantifiers to float between the label l and handle h. Replacing the equalities with qeq constraints in the example above underspecifies scope, giving the MRS shown in (2), and allowing both readings shown in (3).

Robust Minimal Recursion Semantics (RMRS) is a modified MRS representation that in addition to underspecification of scope allows underspecification of relational information (Copestake, 2007). Namely, in RMRS, arguments (ARGs) are represented as distinct elements and can be omitted or underspecified.

The transformation process between MRS and RMRS splits off most elementary predicate arguments and uses anchors (denoted with `a`) to relate EPs and ARGs. For example, `l8:_accept(e2, _, x)` from (2) becomes `l8:a4:_accept(e2)` and `ARG2(a4, x)`. Notice that `ARG0` of each EP is not split off. The `ARG0` specifies a unique relationship between a variable and its associated EP. For example, in (2), variable `x` is uniquely associated with EP `_definition(x, y)`. The full RMRS of (2) is shown in (5):

(5) **LTOP**: `l2`,
`l4:a0:_no, BV(a0, x), RSTR(a0, h7), BODY(a0, _), h7` $=_q$ `l8`
`l2:a1:_exist(e4), ARG1(a1, x)`
`l2:a2:_yet(e5), ARG1(a2, e4)`
`l8:a3:_general(e1), ARG1(a3, e2)`
`l8:a4:_accept(e2), ARG2(a4, x)`
`l8:a5:_formal(e3), ARG1(a5, x)`
`l8:a6:_definition(x), ARG1(a6, y)`
`l5:a7:udefq, BV(a7, y), RSTR(a7, h6), BODY(a7, _), h6` $=_q$ `l3`
`l3:a8:_algorithm(y)`

3.1.2 Dependency Minimal Recursion Semantics

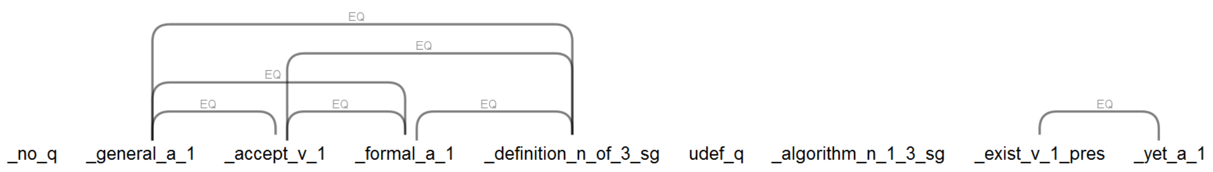
A DMRS representation of a sentence is a directed acyclic graph (DAG) with elementary predicates as nodes. It is constructed from a RMRS representation by combining three subgraphs: (1) label equality graph, connecting EPs with shared labels; (2) handle-to-label qeq graph, connecting handles and labels; and (3) variable graph, connecting EPs with their arguments. I describe each in turn below.

As described in the previous section, two (or more) elementary predicates sharing a label indicates a conjunction between them. For example, in (5), four EPs share the label `l8`. The **Label equality graph** shows the conjunction relationships between nodes representing elementary predicates. The label equality graph for RMRS in (5) is shown in Figure 3.1a. Notice that a conjunction between n EPs results in $n(n-1)/2$ undirected edges (Copestake, 2009).

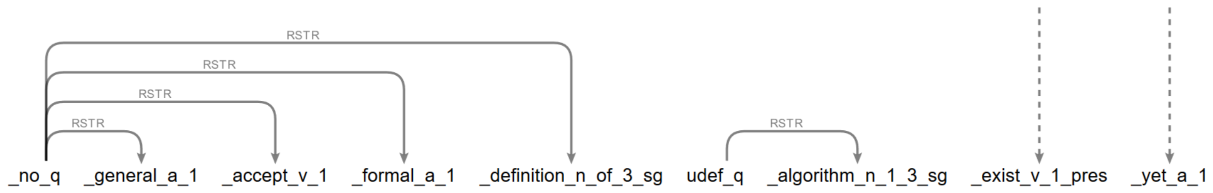
The second graph is the **hole-to-label qeq graph** indicating the qeq constraints between holes and labels. Given a qeq constraint between a hole and a label, a directed edge connects the node with a hole to a node with the specified label. If several nodes share the label, a directed edge is created to each of them. In (5), there are two qeq constraints. Its hole-to-label qeq graph is shown in Figure 3.1b.

The final graph is the **variable graph**, which shows the argument relationships between nodes representing EPs. An argument relationship between an EP and a variable is represented as a directed edge between the EP and variable's uniquely associated EP (via its ARG0, described in Section 3.1.1). For example, `ARG1(a5, x)` specifies a directed edge, labelled with `ARG1`, between `l8:a5:_formal(e3)` and EP associated with variable `x`, which is `l8:a6:_definition(x)`. The variable graph of (5) is shown in Figure 3.1c.

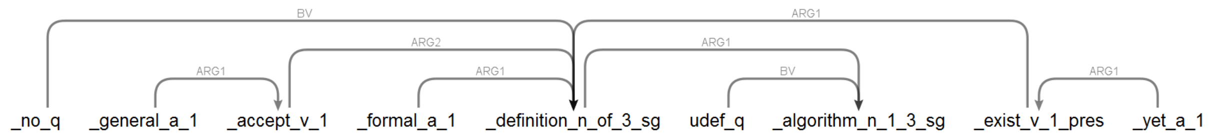
The DMRS graph is constructed by combining the three graphs described above. As label equality and hole-to-label qeq graphs contain redundant edges, a deterministic method is used for selecting the edges to keep in the final DMRS representation. Since the variable graph does not contain redundancies, it is used to guide the selection:



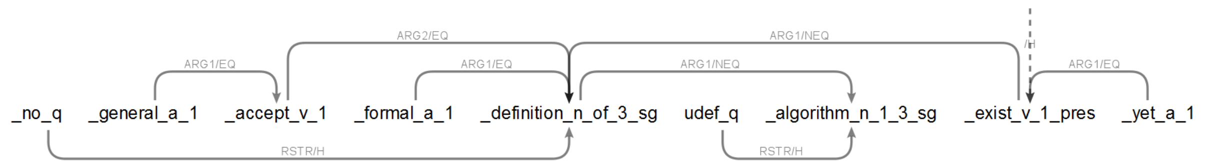
(a) DMRS label equality graph.



(b) DMRS handle-to-label qeq graph.



(c) DMRS variable graph.



(d) Final DMRS graph after merging separate graphs.

Figure 3.1: Construction of DMRS graph for sentence "No generally accepted formal definition of algorithm exists yet." from MRS representation. First, three graphs are constructed from MRS: (a) label equality, (b) handle-to-label qeq, and (c) variable graphs. The final DMRS graph, shown in (d), is the result of combining the three graphs and deterministically removing redundant edges.

1. If the merged graph contains both a label equality and variable edge between a pair of nodes, the two edges are combined. The label of the combined edge has the form `ARG/EQ` (for example, two edges between `_formal_a_1` and `_definition_n_of_3_sg` become one, labelled as `ARG1/EQ`). Note that since label equality edges are undirected, the resulting edges maintain direction of variable edges.

2. Remaining label equality edges are tested for transitivity and removed if found to be redundant. This removes most remaining label equality edges. The exceptions yield undirected `/EQ` edges, which I address in Section 3.2.3.

3. Variable graph edges that do not have a matching label equality edge are labelled as `ARG/NEQ` (for example, `ARG1/NEQ` between `_definition_n_of_3_sg` and `_algorithm_n_1_3_sg`).

4. Hole-to-label qeq and variable graph edges are combined and labelled as `ARG/H`. Redundant hole-to-label qeq edges are removed by selecting a head among nodes

sharing the same label and removing all edges that do not connect to it (for example, `RSTR/H` edge is kept between `_no_q` and `_definition_n_of_3_sg`, but not between `_no_q` and other nodes).

Merging of the three graphs and removing redundant edges results in the final minimally-redundant DMRS graph. The DMRS graph for our example is shown in Figure 3.1d.

3.1.3 MRS in English Resource Grammar and parsing

Minimal Recursion Semantics was primarily designed to be used with grammars developed within a typed feature structure formalism, but is not specific to it (Copestake et al., 2001). MRS is therefore framework-independent. For example, MRS could be used with categorial grammars encoded in feature structures (Copestake et al., 2005). However, MRS is most often used with a Head-driven Phrase Structure Grammar. Head-driven Phrase Structure Grammar (HPSG; Pollard and Sag (1994)) is a heavily-lexicalized unification-based grammar, which uses typed feature structures to encode syntactic information. The English Resource Grammar (ERG, Flickinger (2000)) is a broad-coverage grammar of English written in the HPSG framework. Copestake et al. (2005) define how MRS can be represented as a feature structure and used in ERG as a semantic representation, alongside other feature structures representing syntax.

An HPSG feature structure representation of a sentence is created using a parser employing unification to combine lexical and phrasal feature structures. MRS representation of a sentence is therefore constructed compositionally in parallel with the syntactic analysis of a sentence. I use the ACE parser (Packard, 2016a) with the ERG in order to parse sentences for translation and realization tasks. Parsing settings and parsing coverage statistics are described in experimental setup sections in Chapters 6 and 7.

During parsing, the parser keeps track of the parts of the sentence covered by individual predicates. Therefore, each predicate in an MRS (see Section 3.1.1), and consequently each DMRS node (see Section 3.1.2), has an associated character span in the original (source) sentence. Later in the chapter, I use predicate character spans in order to obtain source alignments in order to construct training examples (see Section 3.3.1).

3.2 DMRS graph modelling

MRS (and consequently DMRS) was designed to include all semantically related information that can be derived from syntax and morphology. However, not all information presented in a representation is useful for its applications. Moreover, some information may harm system performance. Consequently, MRSs are often transformed for use in applications in an application-specific way (for example, Herbelot and Copestake (2006); Bond et al. (2011); Packard (2014)). The same is true for DMRS graphs used for translation and realization with statistical machine translation approaches presented in this thesis. In this section I therefore describe three methods that are used to modify a DMRS graph in order to make it more suitable for algorithms presented later in the thesis, including rule extraction (Section 4.2), rule application (Section 5.3), and decoding (Section 5.4). These methods, by design, result in DMRS graphs that cannot be converted back into MRS.

In Section 3.2.1 I introduce node and edge labels which condense semantic features associated with each node and edge into a single label. Node and edge labels are used by subsequent algorithms (including the ones listed above) as the definition of the graph. Some node semantic features are discarded when creating a label as they are not useful for translation and realization tasks and their inclusion would result in more fine-grained labels, leading to sparsity of the extracted grammar.

In Section 3.2.2 I introduce grammar predicate filtering that removes certain types of grammar predicate nodes from DMRS graphs that do not contribute information useful for translation and realization. Reducing the size and complexity of the graph by removing these nodes helps both decoding and rule extraction algorithms in terms of computational efficiency and grammar sparsity respectively.

Finally, in Section 3.2.3 I introduce graph cycle removal that breaks cycles in the underlying undirected DMRS graph. Although rule extraction and rule application algorithms can operate on graphs with such cycles in them, they are not able to decompose these cycles into smaller units. Therefore, I describe a cycle breaking algorithm and accompanying heuristics that break the cycles in the underlying undirected graph by choosing to remove an edge whose removal will cause the least semantic information to be lost.

3.2.1 Node and edge labelling

Each node in a DMRS graph (introduced in Section 3.1) is associated with many pieces of information. For example, a node representing a noun predicate may include information about predicate lemma, part-of-speech class, predicate sense, person, and number. Algorithms presented in subsequent chapters need an efficient way of comparing whether two nodes are the same. In the case of rule application (presented in Section 5.3), knowing whether a node of a rule in a grammar is the same as the node of the input sentence is the first step in deciding whether a rule can be used to decode the given input graph. In order to efficiently compare nodes and edges, I introduce node and edge labels, which enable direct comparison of strings (or integers if encoded as ids).

The information included and excluded from a label has an important effect on grammar quality. For example, including every piece of information associated with a node in a label will result in a grammar that does not generalize well. Such a grammar will contain a large number of rules that apply to few input DMRS graphs, causing parts of input graphs to be undecodable. On the contrary, excluding important information from the label will result in a grammar that generalizes too much. The resulting grammar will contain fewer rules that apply to many input DMRS graphs, causing incorrect decoding of those graphs. Consequently, information included in node and edge labels is of vital importance and was chosen in order to be suitable for translation and realization tasks.

In the first part of this section I will describe the available node information and the node label composition. The final part of the section will define edge labels.

The available DMRS node properties are presented in Table 3.1. I recognize two distinct classes of nodes: real predicate and grammar predicate nodes. The general label composition for real predicate nodes is:

`_lemma_pos_sense`

Property	**Description**
Predicate	
lemma	Predicate lemma, for example 'stand' for verb 'standing'.
pos	Part-of-speech class, for example verb, noun, adjective/adverb, preposition, and quantifier.
sense	Predicate sense, for example 'for' for verb 'standing (for election)'.
gpred	Indicating a grammar predicate.
carg	Constant argument, stores predicate symbols of numbers, named entities, etc., which are not stored in the lexicon. For example, '56' and 'Mark'.
Nominal features	
gend	Gender, namely 'm', 'f', and 'n'.
ind	Individuated (boolean), corresponds roughly to the notion of countability.
num	Number, namely 'sg' and 'pl'.
pt	Pronoun type, for example standard and reflexive.
pers	Person, namely first, second, third.
Tense, aspect, and mood	
tense	Verb tense, for example present, past, future, and untensed.
perf	Perfective versus imperfective (boolean) aspect.
prog	Progressive versus nonprogressive (boolean) aspect.
mood	Mood and modality, corresponding to opinions/attitudes of the speaker, for example indicative and subjunctive.
Sentential features	
sf	Sentence force, for example proposition, question, and imperative.

Table 3.1: Node property descriptions and their availability across different node types. For some properties additional values exist when they are underspecified, for example 'm-or-f' and 'prop-or-ques'.

Examples of such labels include: `_on_p_temp` (temporal preposition 'on', e.g., 'on Tuesday'), `_illegal_a_for` (adjective 'illegal'), and `_the_q` (quantifier 'the'). Note the absence of sense information in the quantifier example. In the case of missing node properties, they are simply ignored when constructing the label.[1]

The general label composition for a grammar predicate node is:

`carg_gpred`

Examples of grammar predicate labels include: `compound` (joining two parts of a compound), `neg` (denoting a negation), and `8_card` (cardinal number '8').

Below I define more granular labels for some classes of real and grammar predicate nodes:

[1]Although the *general* real predicate label composition is identical to the label composition typically used in Minimal Recursion Semantics, the two should not be confused. The subsequent label compositions discussed in this section differ from the ones used in Minimal Recursion Semantics.

1. **_lemma_pos_sense_pers_num** for noun predicates (pos='n'). The default value of pers and num if not specified is '3' and 'sg' respectively. Examples: _chemistry_n_1_3_sg ('chemistry'), _instrument_n_of_3_pl ('instruments').

2. **_lemma_pos_sense_tense_sf_perf_prog** for verb predicates (pos='v'). The default value for sf is 'prop'. Examples: _fill_v_1_past ('filled'), _happen_v_1_pres_ques (question '(what) happens').

3. **carg_gpred_num** for noun-like grammar predicates. Examples: USA_named_n_sg ('USA'), person_sg (for example corresponding to 'everyone'), winter_season_sg ('winter').

4. **gpred_pers_num_gend** for pronoun grammar predicates. Examples: pron_3_sg_n ('it'), pron_2 ('you'), pron_2_sg ('yourself').

Some of the node properties in Table 3.1 are omitted from these definitions despite their relative importance in English (for example, pronoun type and individuated boolean (pt and ind in Table 3.1)) due to the trade-off between grammar precision and sparsity (discussed above). The node properties used were determined based on their perceived relative importance to translation and realization and small-scale qualitative experiments evaluating their effect.

Compared to node label composition, edge labels are straightforward. DMRS edge labels have two parts: (1) rargname is the argument name (e.g., 'ARG1'), and (2) post represents MRS label equality, qeq, etc. (e.g., 'EQ'). Both properties correspond to the underlying DMRS subgraphs, discussed in Section 3.1.2. The edge label is a composition of the two properties:

rargname/post

Examples of edge labels include: ARG1/NEQ, RSTR/H, and L-INDEX/NEQ.

3.2.2 Grammar predicate filtering

In this section, I introduce grammar predicate filtering. Grammar predicate filtering is the task of removing certain types of grammar predicates which are unlikely to be useful in realization or translation, in order to improve decoding performance and reduce data sparsity. The grammar predicate classes to filter were identified manually and are filtered only when their removal would not result in a disconnected graph.

Grammar predicates, introduced in Section 3.1.1, are a class of over one hundred different predicates occurring in different contexts. They are used by the grammar to convey additional information about lexicalized predicates. For example, a common grammar predicate is compound, which has the purpose of connecting two parts of a proper name or a noun-noun compound (e.g., connecting predicates 'Walter' and 'White' in Figure 3.2); proper_q is a quantifier for proper names when they are not explicitly quantified (e.g., 'Walter White finds cooking fun.' shown in Figure 3.2 versus 'The Walter I saw yesterday.'); nominalization indicates that a non-noun predicate is being used as a noun (e.g., indicating that 'cooking' is used as a noun in 'Walter White finds cooking fun.'); udef_q

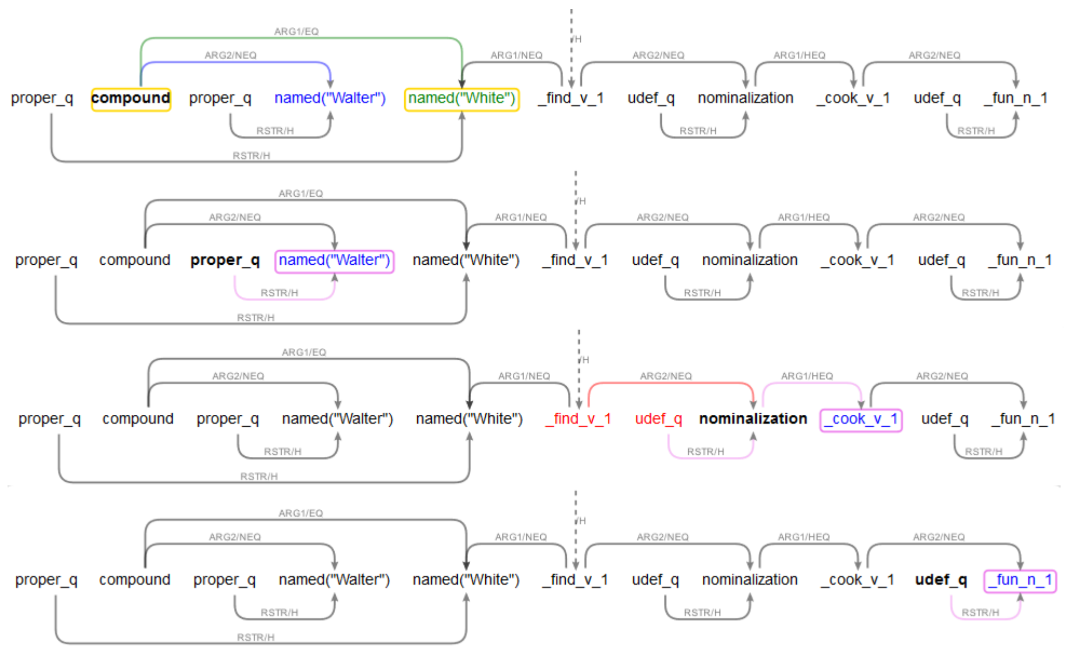


Figure 3.2: Example of `compound`, `proper_q`, `nominalization`, and `udef_q` grammar predicates (gpreds) in the sentence 'Walter White finds cooking fun.'

is a 'placeholder' quantifier in absence of other quantifiers (occurs twice in 'Walter White finds cooking fun.'). The DMRS graph of the example sentence is shown in Figure 3.2. Each of the four grammar predicate examples is highlighted in turn.

Grammar predicate nodes make the DMRS graph larger and introduce additional complexity, which has negative consequences on translation and realization: (1) larger graph size increases the computing resources and time required for decoding (demonstrated in Section 5.7); and (2) additional complexity increases data sparsity in the extracted grammar (described in Chapter 4).

In order to alleviate this problem, I constructed a set of grammar predicates to filter based on their purpose and their value to translation and realization tasks. Most commonly filtered grammar predicates include:

1. `udef_q` - Shown in example above, `udef_q` is inserted by the grammar if there are no other quantifiers present (described in Section 3.1).

2. `pronoun_q` - `pronoun_q` is a quantifier for pronouns.

3. `proper_q` - Shown in example above, `proper_q` is a quantifier for proper names when they are not explicitly quantified.

4. `ellipsis` - Signifies a gap in a sentence (e.g., 'Marie doesn't know when' [...]), but often occurs due to a misparse.

As can be seen from their descriptions, these predicates either do not add information that could be used for realization or translation or they introduce additional noise in the data that has the potential to harm realization or translation. The manually constructed set of grammar predicates to filter includes 30 out of 108 types of grammar predicates.

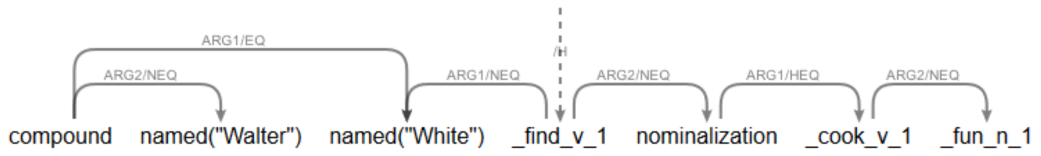


Figure 3.3: DMRS of sentence 'Walter White finds cooking fun.' after grammar predicate filtering. Note the reduced size and simplified structure.

I list the full set of filtered grammar predicates in Appendix A.1. Below I describe the algorithm used for filtering.

FilterGpred(G, F) is an algorithm that removes grammar predicates from graph G if they occur in the filter set F. **Alg. 3.1**

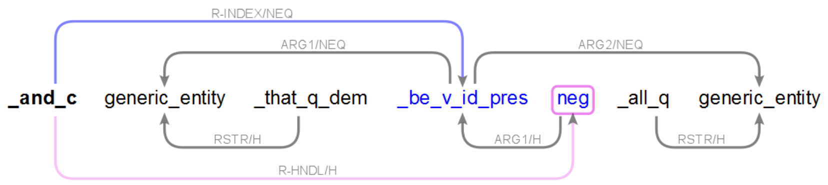
1. (Initialize the list of nodes to filter.) Set $N_F \leftarrow [\,]$.
2. (Find nodes to filter.) For each node n_i in G, append n_i to N_F if n_i is a gpred node and its gpred type is in F.
3. (Remove nodes from graph.) For each node n_i in list N_F, test if its removal would create a disconnected graph. If it does not, remove from graph G node n_i and any edge starting or terminating at n_i.
4. (Return.) Return filtered graph G.

The result of filtering grammar predicates from example in Figure 3.2 is shown in Figure 3.3. Filtering removed 4 grammar predicate nodes (two `udef_q` and two `proper_q` nodes), reducing the size of the DMRS (node set) by 36%. Before filtering, rule application algorithm (described in Section 5.3) extracted 36 different rule source sides for the example sentence. After filtering, it extracts 19, a reduction of 47% which greatly helps the grammar data sparsity. Evaluating its effect on dataset level, filtering reduced the average number of grammar predicates per sentence from 7.25 to 3.47 (evaluated on 1774 sentences of filtered newstest2013 dataset, introduced in Chapter 6).

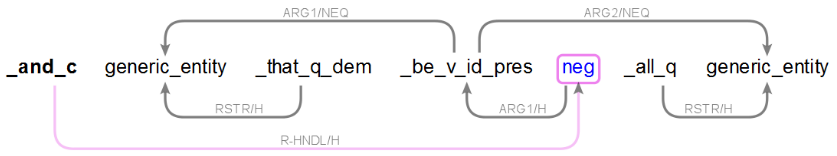
3.2.3 Graph cycle removal

Semantic representations, including DMRS and AMR representations (Banarescu et al., 2013), often model sentence semantics as a directed acyclic graph. However, in addition to DMRS representations being directed acyclic graphs, they are often polytrees. A polytree is a directed acyclic graph whose underlying undirected graph is a tree (Rebane and Pearl, 1987). That is, many DMRS graphs would be acyclic even if their edges were undirected. In DMRS representations, cycles in underlying undirected graphs arise due to the variable graph, described in Section 3.1.2 (the exception are cycles caused by undirected `/EQ` edges, discussed below). They are often used to model specific phenomena (such as control).

The polytree property of DMRS graphs is important because of the requirements of rule extraction and rule application algorithms described in Sections 4.2 and 5.3. These algorithms can operate on directed acyclic graphs, but are not able to decompose the cycles



(a) Conjunction index/handle cuug. Note that outgoing edges `R-INDEX/NEQ` and `R-HNDL/H` of node `_and_c` form a cuug via `_be_v_id_pres` and `neg` nodes.



(b) By removing the `R-INDEX/NEQ` edge, the cuug is broken.

Figure 3.4: An example of conjunction index/handle cuug in sentence "And that's not all.".

in underlying undirected graph into more fine-grained rules. Therefore, it is beneficial to make every DMRS graph a polytree.

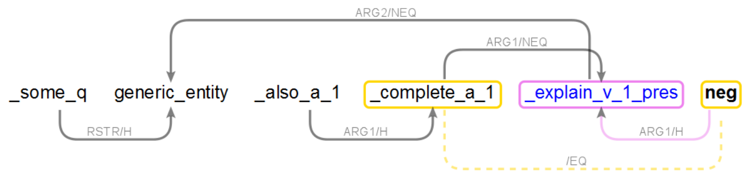
In this section, I describe the approach for converting DMRS directed acyclic graphs into polytrees. The algorithm recognizes the most common phenomena that cause cycles in the underlying undirected graph and breaks them by removing an edge. These phenomena are targeted with individual heuristics in order to remove as little semantic information from the DMRS graph as possible. A general cycle breaking strategy removes the remaining cycles. In the remainder of this section, I will refer to a cycle in the underlying undirected graph as **cuug**. Below, I show the examples of five most common phenomena causing cuugs in DMRS and describe the heuristic for breaking them.

Firstly, **conjunction index/handle cuug** occurs with conjunction nodes which have one or two pairs of outgoing edges, for example: `R-INDEX/NEQ` and `R-HNDL/H`, and `L-INDEX/NEQ` and `L-HNDL/H`. Pairs of edges essentially occur due to notation overloading in the grammar, where a single type of conjunction node may be used in multiple contexts. An example of a cuug when one such pair connects to two different nodes is shown in Figure 3.4a. The cuug breaking heuristic removes the `INDEX` edge, for whichever pair of edges is part of the cuug. In the example, `R-INDEX/NEQ` edge is removed, making the resulting graph shown in Figure 3.4a acyclic.

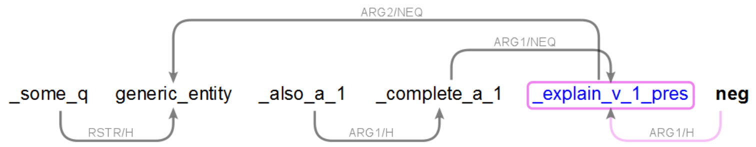
EQ cuug occurs due to the presence of an undirected `/EQ` edge (see Section 3.1.2) in the DMRS graph. The heuristic for breaking the cuug removes the `/EQ` edge. An example of an EQ cuug is shown in Figure 3.5a, while its acyclic version is shown in Figure 3.5b.

Control is a construction in which the subject of a non-finite clause is understood to be the same as the controller. Take sentence "Singapore seeks babies to save its economy" as an example. 'Singapore' is the controller, that is the understood subject, of the non-finite clause 'save its economy'. In DMRS, a control construction is represented with a cuug. The DMRS of the example sentence is shown in Figure 3.6a. The cuug is broken by removing the edge between the non-finite clause predicate (`_save_v_1`) and the controller subject (`Singapore_named_sg`). The resulting acyclic graph is shown in Figure 3.6b.

Small clause is a construction similar to control, which has a clause without an overt verb. Like control, small clause constructions are represented with cuugs in DMRS. An example of a small clause DMRS is shown in Figure 3.7a. The cuug breaking heuristic

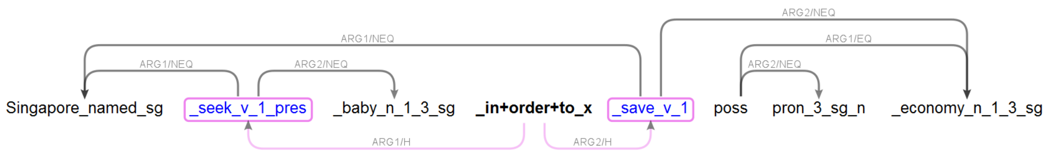


(a) EQ cuug. Note that `/EQ` edge forms a cuug via `_complete_a_1`, `_explain_v_1_pres`, and `neg` nodes.

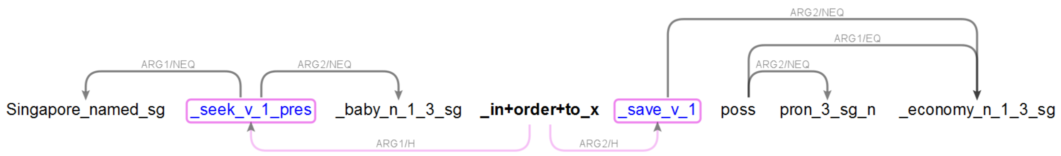


(b) By removing the `/EQ` edge, the cuug is broken.

Figure 3.5: An example of EQ cuug in sentence “Some are also completely unexplained.”.

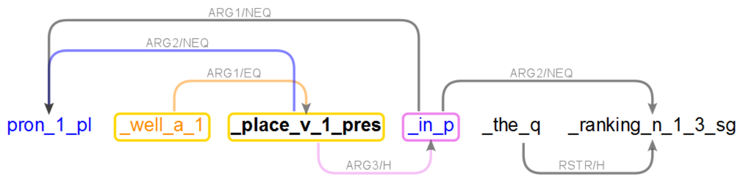


(a) Control cycle between nodes `_in+order+to_x`, `_save_v_1`, `Singapore_named_sg`, and `_seek_v_1_pres`.

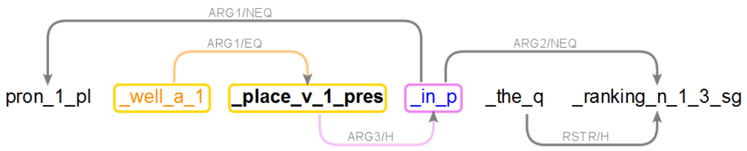


(b) By removing the `ARG1/NEQ` edge between `Singapore_named_sg` and `_save_v_1`, the cuug is broken.

Figure 3.6: An example of control cuug in sentence “Singapore seeks babies to save its economy”.

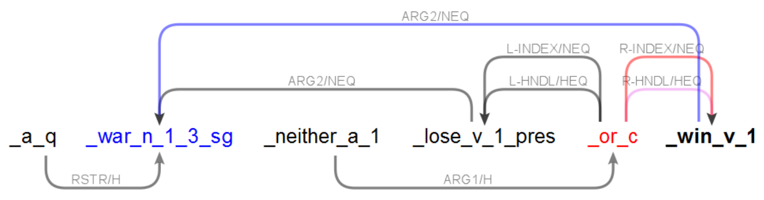


(a) Small clause cuug between nodes `_place_v_1_pres`, `_in_p`, and `pron_1_pl`.

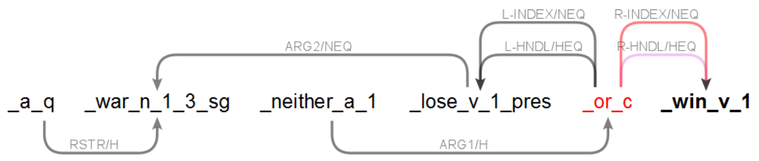


(b) By removing the `ARG1/NEQ` edge between `_in_p` and `pron_1_pl`, the cuug is broken.

Figure 3.7: An example of small clause cuug in sentence “We’re well placed in the ranking.”.



(a) Verb or adjective conjunction cuug between nodes `_or_c`, `_win_v_1`, `_war_n_1_3_sg`, and `_lose_v_1_pres`.



(b) By removing the longer edge `ARG2/NEQ` edge between `_win_v_1` and `_war_n_1_3_sg`, the cuug is broken.

Figure 3.8: An example of verb or adjective conjunction cuug in sentence "A war that is neither lost or won".

removes the edge between the matrix verb and its subject. The resulting example graph is shown in Figure 3.7b.

Finally, **verb or adjective conjunction cuug** occurs with a conjunction of two verbs or adjectives referring to the same subject. The cuug is broken by removing one of the two edges between the verbs/adjectives and the subject. In the case of translation, the longer edge (spanning most aligned source tokens, described below) of the two is removed. In the case of realization, a random choice is made between the two. A different strategy is used for realization because source tokens are not available (this corresponds to the use of different default breaking strategies between the tasks, explained below). An example with a conjunction of two verbs is shown in Figure 3.8a and resulting acyclic graph in Figure 3.8b.

The five heuristics described above are used by the graph cuug removal algorithm to iteratively remove cuugs from a graph. The algorithm is described below.

Alg. 3.2 **CuugRemoval**(G) is an algorithm that removes cycles from the underlying undirected graph of G to make G a polytree.

1. (Remove cuugs until none are left.) While G contains a cuug:
    (a) Detect type of cuug in the following order: (1) conjunction index, (2) eq edge, (3) control, (4) small clause, and (5) verb or adjective conjunction.
    (b) When a cuug type is detected, attempt to break it using the corresponding heuristic (described above).
    (c) If none of the above types of cuug are detected, break it with the default cuug breaking strategy.
    (d) After the cuug is broken, execution proceeds to another loop iteration.
2. (Return.) Return G with no cuugs in underlying undirected graph.

The five cuug types listed in statement (1a) correspond to the five cuug breaking heuristics discussed above. The remaining cuugs are broken with a default cuug breaking strategy. The default strategy differs between translation and realization approaches. In the case of translation, all edges forming the cuug are collected and the longest edge is removed in order to break the cuug. For graph $G = (V, E)$, I define edge length as:

$$\text{len}(e) = \min_{t_i \in \mathcal{A}(v_i), t_j \in \mathcal{A}(v_j)} |\text{index}(t_i) - \text{index}(t_j)| \tag{3.1}$$

where v_i and v_j are elements of V and incident to edge $e, e \in E$. $\mathcal{A}(v)$ returns the set of tokens aligned to node v and index(t) returns the position of token t in the source sentence. The alignment of nodes to source sentence tokens is discussed in Section 3.3.1. For now, suffice to say that each node can be aligned to multiple tokens of the source sentence. The edge length is therefore computed as the minimum token distance between a pair of source tokens, each aligned to one of the vertices connected by the edge.

The translation default cuug breaking strategy is a proxy for preserving local semantic information in favour of semantic dependencies between more distant parts of the sentence. However, the source sentence token information is not available in realization, where the starting point is the DMRS graph on its own. A simplified default cuug breaking strategy for realization considers all edges forming the cuug and breaks the edge going to the most modified node. For graph $G = (V, E)$, I define node modification count as:

$$\text{mod\_count}(v) = \left|E^{-}(v)\right| \tag{3.2}$$

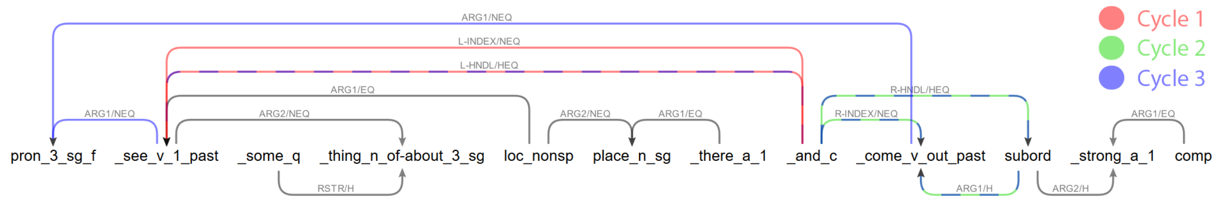
where $E^{-}(v)$ denotes the set of incoming edges to node $v, v \in V$. The target node modification count resembles the edge length as more modified nodes will tend to be further away in the original source sentence. The goal is therefore again to preserve local semantic information in favour of semantic dependencies between more distant parts of the sentence.

In Table 3.2 I show the statistics of cuug occurrence on a set of 2830 sentences. Around half of sentence DMRS graphs contain at least one cuug. The most common phenomenon are EQ cuugs, occurring in 28.6% of all sentences and accounting for 41.1% of all cuugs. The five heuristics for specific phenomena break 82.9% of all cuugs; the remaining 17.1% cuugs are broken with the default strategy.

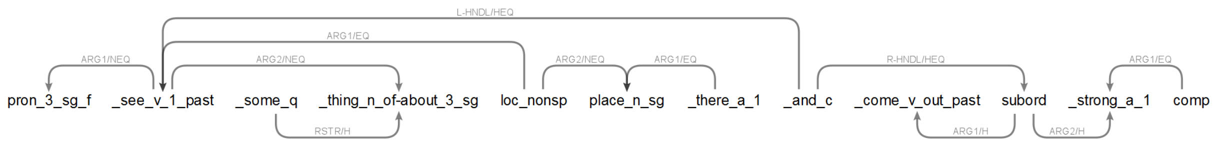
Finally, I demonstrate the result of cuug breaking algorithm on a DMRS of sentence "She saw some things there and came out stronger." shown in Figure 3.9a. The DMRS graph contains three cuugs shown in different colours. The algorithm first breaks cuugs 1 and 2 by detecting them to be conjunction index cycles and using the corresponding heuristic. This removes edges `L-INDEX/NEQ` between nodes `_and_c` and `_see_v_1_past`, and `R-INDEX/NEQ` between nodes `_and_c` and `_come_v_out_past`. The third cuug is a verb or adjective conjunction cuug with two verbs referring to the same subject. However, the cuug cannot be broken with the corresponding heuristic because of the `subord` node preventing its detection. Therefore the final cuug is broken with the default cuug breaking strategy, which removes edge `ARG1/NEQ` between nodes `_come_v_out_past` and `pron_3_sg_f`, based on its distance between the corresponding tokens. The resulting acyclic graph is shown in Figure 3.9b.

cuug type	% sentences	% cuugs
Conj. index/handle	9.6	11.8
EQ	28.6	41.1
Control	11.8	14.5
Small clause	4.0	4.6
Verb or adj. conj.	9.4	10.9
Other (default)	12.6	17.1
All	49.2	100.0

Table 3.2: Occurrence of cuugs types on a set of 2830 sentences (newstest2013 described and used in Chapters 6 and 7 for tuning, consisting of 3000 sentences, 170 of which did not produce a parse). Sentence column gives a percentage of sentences containing at least one cuug of a given type. Cuug column gives the proportion of cycles of that type out of a total of 2,496 detected (and broken) cuugs.



(a) DMRS graph before cycle removal with three cuugs, shown in red, green, and blue.



(b) DMRS graph (polytree) without cuugs, after three edges are removed.

Figure 3.9: DMRS graph for sentence "She saw some things there and came out stronger." (a) before and (b) after cycle removal.

3.3 Alignment

In statistical machine translation, alignment refers to the problem of determining the links (alignments) between words in source language and words in target language that are translation equivalent. SMT alignment usually forms the first step of training an SMT system and is required for extracting a phrase table or a grammar. In the approach presented in this thesis, rule extraction (described in Section 4.2) is a procedure used for extracting rules from pairs of aligned DMRS graphs and tokenized sentences. In this section I therefore consider the problem of aligning nodes of a DMRS graph to sentence tokens.

In Section 3.3.1 I describe the problems of and solutions to source alignment, which is concerned with aligning DMRS nodes to a tokenized source sentence. I additionally discuss the issues of punctuation. In Section 3.3.2 I describe an approach to obtain source-target alignments, which are necessary for translation rule extraction. Source-target alignments, between DMRS nodes and target sentence tokens, are obtained by extending source alignments to target tokens by using SMT alignments. Finally, in the background Section 3.3.3,

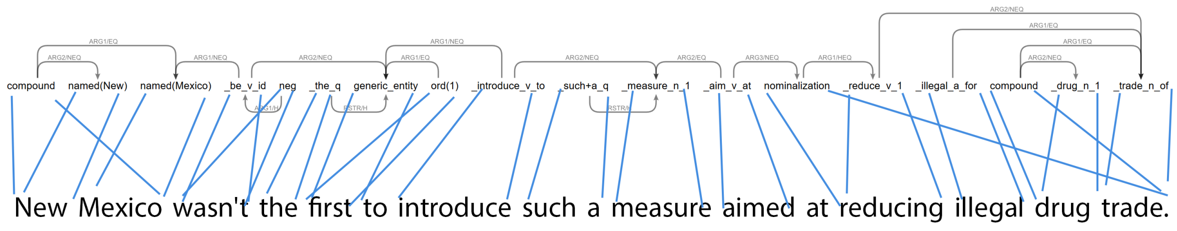


Figure 3.10: Alignment between DMRS and untokenized source sentence as obtained by ERG parsing procedure. Note that each alignment is a character span over the sentence.

I describe the problem of alignment in statistical machine translation and common approaches used to solve it.

3.3.1 Source alignment

Source alignment refers to the problem of determining alignments between DMRS nodes and tokens of the source sentence (i.e., the sentence that was parsed to obtain the DMRS graph). In the case of realization, source alignments are used directly for rule extraction; in the case of translation, they are extended to target side via SMT alignments (described in Section 3.3.2).

Nodes of a DMRS graph include partial information about their alignment to the source sentence already. Namely, the character range of the untokenized source sentence is known for each node of the graph, obtained as a by-product of the ERG parsing procedure (introduced in Section 3.1.3). In order to obtain full alignments to source sentence tokens, three problems need to be solved: (1) misalignment between ERG-style tokenization and Penn Treebank (PTB) style tokenization; (2) unaligned source tokens; and (3) excessive grammar predicate alignments. Below, I address each of the problems in turn. I conclude the section with a discussion of punctuation.

An example of the ERG character range alignments to an untokenized sentence is shown in Figure 3.10. The character ranges of each node over an untokenized source sentence implicitly define an ERG tokenization of the source sentence. As it is evident from the example figure, ERG-style tokenization differs from the more common PTB-style tokenization in several ways, as discussed by Fares (2013):

1. Most punctuation marks are treated as pseudo-affixes to words instead of being tokenized off as it is common in PTB-style tokenization. This can be seen in the alignment of `_trade_n_of` node to string 'trade.'

2. Hyphens and slashes can introduce token boundaries (e.g., 'open-', 'source').

3. Contracted negations are not tokenized off. We can observe this in the alignment of nodes `_be_v_id` and `neg` to the same string 'wasn't '.

4. ERG treats some multi-word expressions as a single token (e.g., 'ad hoc').

Although ERG-style tokenization has its strengths (Fares, 2013), it is not well suited to the domain of statistical machine translation. For example, consider the token 'trade.'

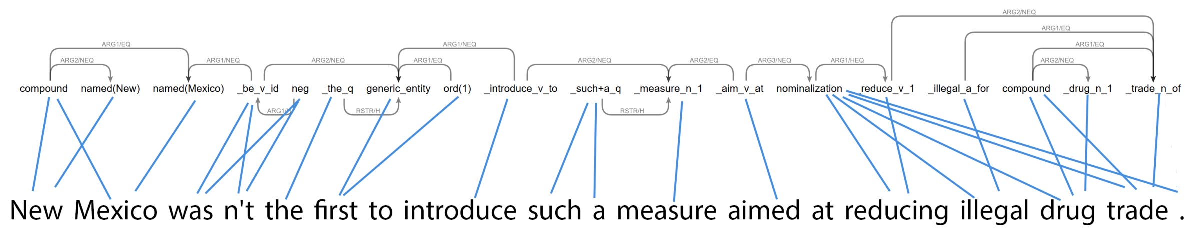


Figure 3.11: Alignment between DMRS and PTB-style tokenized source sentence, obtained by relating provided ERG-style tokenization to PTB-style tokenization. Note that each alignment is a single link between a node and a token.

listed as an example above. In order to translate such a token using SMT methods, an exact occurrence of it needs to be seen in the training data. Even in the case that a sentence ending in ‘trade.’ occurs in the training data, it is likely that the aligned target string, for example ‘Handel’ in German, will not occur at the end of sentence. The aligned token will therefore not include the full stop and the extracted rule will delete it altogether. PTB-style tokenization is more suitable to the domain of SMT, as it separates the full stop into a standalone token, allowing for easier token reordering between the two languages and extracting fewer rules. Despite these disadvantages, ERG tokenization has features that could benefit translation: splitting of hyphenated words and treating multi-word expressions as a single token. In the solution I describe below, these features are preserved.

The goal of source alignment is aligning DMRS nodes to a PTB-style tokenized source sentence. In the first step, the alignments are created using the existing character spans associated with each node. I relate the two styles of tokenization with an algorithm that first matches nodes that align to a single token and then uses that information to align nodes with multiple tokens. The algorithm’s token match between the two tokenization style is achieved without respect to whitespace, capitalization, and punctuation differences. An additional degree of difference between the two tokens is allowed with the use of the Levenshtein distance. The result of this algorithm for the example introduced above is shown in Figure 3.11.

Note, however, that in Figure 3.11 some tokens, namely ‘to’ and ‘at’, are not aligned to any node. In fact, they were not included in any node’s character span in Figure 3.10 either. These tokens, referred to as null semantics items or contentless lexical entries, as their name implies do not carry semantic meaning on their own and are therefore not aligned to any node (however, contentless lexical entries may contribute to semantics of other predicates, for example influencing their tense, e.g., auxiliary *have*). Their treatment in MRS is in line with the usual practice in formal semantics (Copestake et al., 2005).

However, regardless of their paucity of meaning, a realization system needs to realize such words alongside semantically meaningful ones in order to produce fluent output; were unaligned words missing, source-target alignment for translation described in Section 3.3.2 would fail to create alignments necessary to produce equivalent words on the target side. Therefore, it is necessary to find suitable alignments for all unaligned tokens. I describe a heuristic approach below. Note that ERG faces the same problem (Carroll et al., 1999) and that our heuristics were initially developed from corresponding rules found in ERG (known as trigger rules).

The approach uses heuristics associated with most common unaligned token strings to

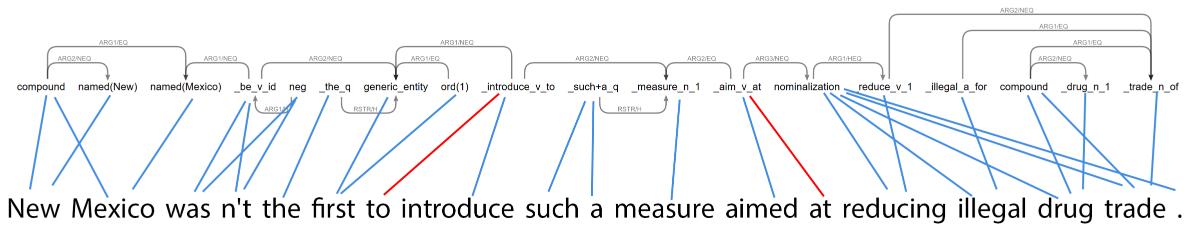


Figure 3.12: Alignment between DMRS and PTB-style tokenized source sentence, additionally extended with alignment to previously unaligned tokens (shown in red).

decide which node the unaligned token should be aligned to:

1. If an applicable heuristic function exists for an unaligned token (for example, a heuristic function for aligning *do*), it is used to determine its alignment. If no applicable heuristic functions exist, the token remains unaligned.

2. Adjacent unaligned tokens are considered first as a unit (for example, *has been*, *much rather*), before they are considered as individual tokens.

3. A heuristic alignment function attempts to align a token to a node, which is already aligned to tokens to its left or right. An alignment is created if the node matches the set of criteria defined by the function. For example, the heuristic function for aligning token 'do' attempts to align it to a verb node in present tense, aligned to the nearest token to its right; the heuristic function for aligning 'much rather' attempts to align it to a conjunction node with lemma 'rather+than', aligned to the nearest token to its right.

The heuristic alignment functions were manually defined in an iterative process where I qualitatively evaluated their performance on a set of sentences. The result of aligning previously unaligned tokens of the example introduced above is shown in Figure 3.12.

The final problem of source alignment relates to excessive grammar predicate alignments. In rule extraction, alignments constrain which group of nodes (i.e., subgraph) aligned to a sequence of nodes can be extracted as a rule (detailed description is provided in Section 4.2). Consequently, when a node is aligned to a large number of tokens, it harms the ability of the rule extraction algorithm to extract effective rules.

An example of such a node can be seen in Figure 3.12 with `nominalization`. As described in Section 3.2.2, a nominalization grammar predicate indicates that a non-noun predicate is being used as a noun. As such, it is aligned to the sequence of tokens 'reducing illegal drug trade .' Due to constraints imposed by its alignments, rules for token sequences 'trade', 'drug trade', and 'illegal drug trade' cannot be extracted by the rule extraction algorithm.

The type of alignment behaviour we observed above only occurs with certain types of grammar predicates. As a somewhat crude solution, I limit the maximum number of tokens that a grammar predicate can be aligned to to three.[2] If a grammar predicate aligns to more than three tokens, all of its alignments are removed. The result of this

[2]The maximum number of tokens that a grammar predicated can be aligned was determined qualitatively in a small experiment of 100 sentences.

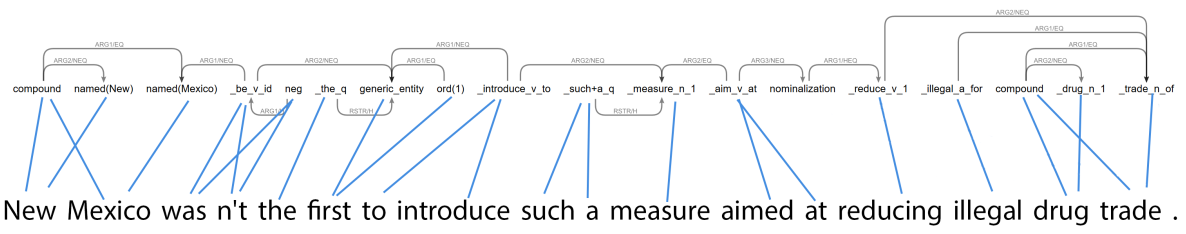


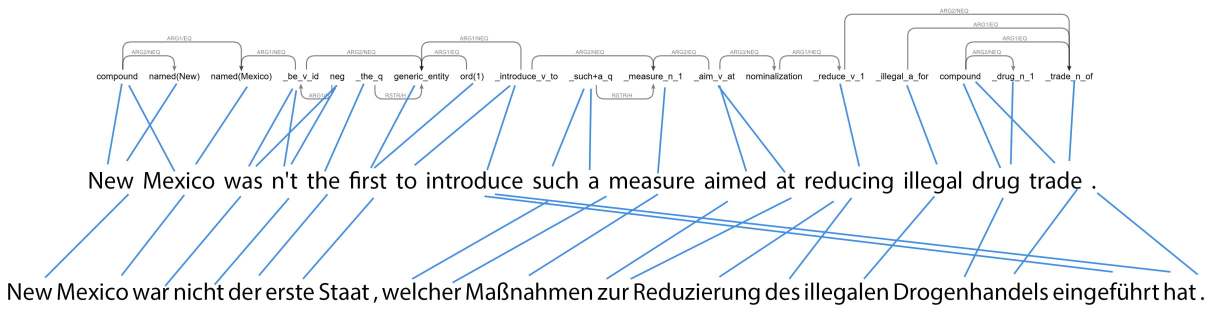
Figure 3.13: Alignment between DMRS and PTB-style tokenized source sentence, with removed alignments for the grammar predicate `nominalization`.

alignment filter for our example is shown in Figure 3.13 and represents the final state of source alignments for the example sentence.

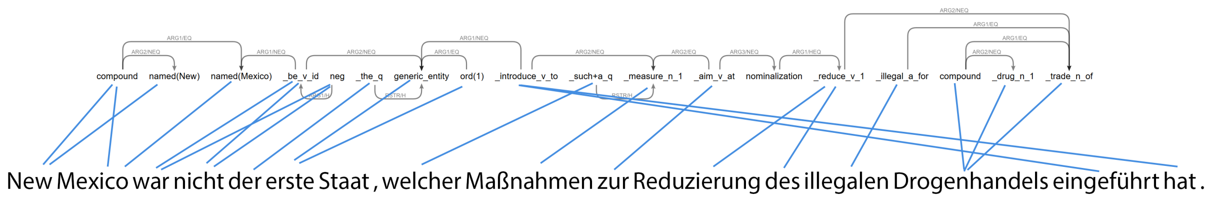
Finally, I conclude this section with a discussion on the topic of punctuation. In the description of the first problem of source alignment, the mismatch between ERG and PTB-style tokenization, I noted the difference in how the two tokenization styles handle punctuation. Punctuation is not explicitly represented in DMRS graphs in the form of nodes and is instead attached as a pseudo-affix to the preceding token (a discussion on punctuation in ERG can be found in Adolphs et al. (2008)). As discussed above, such handling of punctuation is undesirable for rule extraction in most SMT systems as it harms extraction of rules in general. Consequently, the solution to the first problem ignores punctuation when matching between tokens in the two tokenization styles, leaving the punctuation tokens unaligned. This can be seen in the final aligned Figure 3.13, where the full stop at the end is not aligned to any node.

Handling of punctuation in such a way has the downside of extracting fewer rules dealing with punctuation and negatively impacts the performance of the system. However, unaligned (punctuation and other) tokens occurring in between aligned tokens are still extracted as part of the encompassing rule, meaning that in many cases internal punctuation is dealt with sufficiently. The bigger problem are punctuation tokens at the end of the sentence, for many of which no rules are extracted. A simple solution to this problem is a post-processing step which attaches end punctuation accordingly.

Additionally, I explored an alternative approach to dealing with punctuation, which was used in Horvat et al. (2015). In the alternative approach, the DMRS graph was augmented with punctuation nodes, which were attached to the nodes aligned to the preceding token in the source sentence. The approach treads the line of using information that should not be available at decoding time - attaching punctuation nodes to existing nodes based on source sentence information - but can be seen as a post-parsing step to make the punctuation information explicit in the graph. The strength of the approach is that it makes the punctuation position in realization very clear and easy to reproduce. The downside of the approach is that it is less useful for translation, as it does not allow for reordering of punctuation tokens in the target language. In fact, punctuation tokens could only appear in a translation next to tokens which correspond to tokens which had punctuation next to them in the source sentence. Due to these limitations, I instead opted for the more straightforward solution described above.


compound
named(New)
named(Mexico)
_be_v_id
neg
_the_q
generic_entity
ord(1)
_introduce_v_to
_such+a_q
_measure_n_1
_aim_v_at
nominalization
_reduce_v_1
_illegal_a_for
compound
_drug_n_1
_trade_n_of
New Mexico was n't the first to introduce such a measure aimed at reducing illegal drug trade .
New Mexico war nicht der erste Staat , welcher Maßnahmen zur Reduzierung des illegalen Drogenhandels eingeführt hat .

(a) Alignment between DMRS and source sentence tokens, and SMT alignments between source and target sentence tokens.



(b) Extended alignment between DMRS and target sentence tokens.

Figure 3.14: Extending source alignments to target sentence tokens using SMT alignments for example in Figure 3.13.

3.3.2 Source-target alignment for translation

Source alignment, described in the previous section, creates alignments between DMRS nodes and source sentence tokens. Source alignments are sufficient to create training examples for realization rule extraction. In order to extract translation rules, however, alignments between DMRS nodes and *target* sentence tokens are needed.

I propose a simple approach that takes advantage of source alignments and of alignments between source and target language sentence *tokens*. The latter are referred to as *word alignments* in statistical machine translation and are obtained using an unsupervised approach such as Expectation Maximization. I describe the word alignment problem and common approaches to it in the following section (Section 3.3.3).

The translation alignments are obtained by extending the source alignments, from DMRS nodes to source tokens, to target tokens via the word alignments. The example from previous section (Figure 3.13) combined with word alignments is shown in Figure 3.14a. The resulting alignments between DMRS nodes and target tokens are shown in Figure 3.14b.

3.3.3 Alignment in SMT

In the previous section, I described how the alignments between DMRS nodes and target sentence tokens are obtained for the translation task training examples. The procedure requires alignments between source and target language sentence tokens, referred to as *word alignments* in statistical machine translation. In this background section, based on the paper by Och and Ney (2003), I describe the word alignment problem and the most common approaches to solving it.

The alignments between two sentences can be complicated. They can include reorderings, omissions, insertions, and one-to-many relationships. Generally, alignment between two sequences of words (in source and target language) can be defined as a subset of the Cartesian product between their word indices. However, word alignments are commonly restricted with the additional condition that each source word is aligned to exactly one target word.

The seminal work on word alignment by Brown et al. (1993) (IBM Models 1-5, described below) models translation process with word alignment as a hidden variable. Their models belong to the class of statistical word alignment approaches. In statistical models, alignment is introduced as a hidden variable to the standard translation probability:

$$Pr(f_1^J \mid e_1^I) = \sum_{a_1^J} Pr(f_1^J, a_1^J \mid e_1^I) \tag{3.3}$$

where an alignment maps source position j to target position a_j and f and e denote source and target words.

Although linguistic knowledge can be considered for word alignment, most commonly used word alignment approaches in SMT are language agnostic. The only requirement is a sentence-aligned bilingual corpus with sentences segmented into words. However, morphology needs to be considered in morphologically rich languages in order for word alignment to be successful (Och and Ney, 2003). The remainder of this section focuses on statistical word alignment models.

The difference between most commonly used statistical models is in how they decompose the probability $Pr(f_1^J, a_1^J \mid e_1^I)$ in Equation 3.3. The **Hidden Markov Alignment Model** decomposes the probability into three parts: length, alignment, and lexicon probabilities. It introduces two assumptions: (1) first-order dependence for alignments (i.e., alignment probability only conditions on one previous alignment), and (2) lexicon probability depends only on the aligned word:

$$Pr(f_1^J, a_1^J \mid e_1^I) = p(J \mid I) \prod_{j=1}^{J} p(a_j \mid a_{j-1}, I) p(f_j \mid e_{a_j}) \tag{3.4}$$

where J and I correspond to length of target and source sentences respectively.

Whereas HMM uses first-order dependence, Models 1 and 2 use zero-order dependence. Model 1 assumes a uniform distribution for alignments (which means that word order does not affect them):

$$Pr(f_1^J, a_1^J \mid e_1^I) = \frac{p(J \mid I)}{(I+J)^J} \prod_{j=1}^{J} p(f_j \mid e_{a_j}) \tag{3.5}$$

In Model 2, word order does affect the alignment probabilities:

$$Pr(f_1^J, a_1^J \mid e_1^I) = p(J \mid I) \prod_{j=1}^{J} p(a_j \mid j, I, J) p(f_j \mid e_{a_j}) \tag{3.6}$$

Models 3, 4, and 5 are fertility-based alignment models with a more complicated structure than Models 1 and 2. Fertility-based models include a probability modelling that a target word is aligned to a certain number of source words (its fertility). Fertility-based models are described using inverted alignments (from target to source), with the requirement that all positions of the source sentence must be covered exactly once (allowing source words to be aligned to the empty word). Models 3-5 define inverted alignment probability distribution with increasing complexity. For brevity, I omit the alignment probability definition of Models 3-5 (an intuitive description and definition can be found in Och and Ney (2003)).

Models 3 and 4 are deficient: the probabilities of all valid arguments do not sum to one. Model 5 is therefore a reformulation of Model 4 with a refined alignment model that avoids deficiency. Och and Ney (2003) point out an interesting similarity between the HMM alignment model and Model 4: "HMM predicts distance between subsequent source language positions, whereas Model 4 predicts distance between subsequent target language positions." In summary, the alignment models described above differ mainly on: (1) alignment model (zero- or first-order dependence), (2) fertility model they use, and (3) whether they are deficient.

Computing the best alignment (Viterbi alignment) has polynomial complexity for Models 1, 2 and the HMM alignment model (Vogel et al., 1996). However, the task is infeasible for Models 3-5. Instead, Model 1 or 2 Viterbi alignment is greedily improved with the refined model in order to obtain pseudo-Viterbi alignment. The statistical model parameters (probabilities) are estimated using maximum-likelihood approach by applying the Expectation Maximisation algorithm (Baum, 1972). The models are usually trained in succession on the same data: final parameters of a previous model are used as a starting point for the next model by weighing its alignments.

Finally, I address the restriction of allowing each source word to align to exactly one word. As observed by Och and Ney (2003), this restriction results in a systematic loss of recall. The solution is to train alignments in both translation directions and combine the two sets of alignments. Two common methods of combining alignments are intersection and union. Union has lower precision but higher recall (the opposite is true for intersection), which is preferable for SMT and is consequently most commonly used.

In this thesis I use the HMM model in both directions, whose alignments are subsequently combined via union, for aligning source and target words (see Section 6.2).

Chapter 4

Rule extraction

A *synchronous context-free grammar* (SCFG) employed by *hierarchical phrase-based translation*(Hiero; Chiang, 2005, 2007) is used in decoding of a previously unseen input as a resource from which rules applicable to the input are drawn from. It serves a similar purpose as a phrase table in phrase-based translation (Koehn et al., 2003). A Hiero SCFG is extracted automatically from a large sentence-aligned parallel corpus. Rules extracted from training examples and conforming to Hiero constraints are subsequently aggregated to form the SCFG. Each training example consists of a source- and target-side sentences and the word alignment between them (statistical word alignment is described in Section 3.3.3). I review the Hiero SCFG and rule extraction in more detail in the background Section 4.1.

Unlike string inputs and outputs of Hiero, the inputs to the statistical machine translation approach discussed in this thesis are DMRS graphs. Consequently, the standard string-based synchronous context-free grammar employed by Hiero and the associated rule extraction algorithms are insufficient for the proposed approach. In the second part of the chapter I describe (1) the adaptation of the string-based synchronous context-free grammar formulation to accommodate DMRS graphs on the source side and (2) the associated rule extraction algorithm.[1] The rule extraction algorithm extracts a set of graph-to-string rules from a single training example, consisting of a source-side DMRS graph, a target-side token sequence, and the alignment between them (see Chapter 3). The rule extraction algorithm proceeds in two stages. It starts by finding semantic subgraphs of the source-side DMRS graph and constructing terminal graph-to-string rules from those subgraphs. It then proceeds to create non-terminal rules by iteratively subtracting terminal rules from other rules. The extracted rules are subject to a set of rule constraints modelled after Hiero rule constraints that ensure that they are faithful to the training example, are useful for translation and realization tasks, and can be used tractably by the decoding algorithms described in Chapter 5. The adapted graph-to-string SCFG and the associated rule extraction algorithm are described in Section 4.2.

The rule extraction algorithm is task-agnostic: it extracts rules for both translation and realization tasks in exactly the same way. The difference between the two tasks is in the provided training example. The source-side DMRS graph is in a different language than the target token sequence for the translation task, whereas they are in the same language for the realization task.

[1]An initial version of the adapted SCFG and the rule extraction algorithm was described in Horvat et al. (2015).

In order to extract a grammar that is capable of producing high quality translations and realizations, rules are extracted from millions of training examples. Extracted rules are subsequently aggregated to form a synchronous context-free grammar. However, graph-to-string rule aggregation poses a challenge. Determining the equality of a pair of graphs is known as the graph isomorphism problem. I address the problem by proposing a heuristic solution to the related problem of graph canonization.

The decoder uses a log-linear model to select the best translation or realization of an input DMRS graph (see Section 5.4.3). The log-linear model relies on rule features in order to make that decision. In addition to the standard set of SMT rule features, I define rule type indicator features based on the DMRS structure in the rules' source-side graphs. I describe grammar construction and rule features in more detail in Section 4.3.

I conclude this chapter by analysing the grammars extracted by the proposed rule extraction and grammar construction approach. I extract a number of translation and realization grammars with a range of training set sizes. I analyse and compare the extracted grammars in terms of their size, rule counts, and rule types. I describe the experiment setup and report the results in Section 4.4.

4.1 Hierarchical phrase-based rule extraction

Hierarchical phrase-based translation (also referred to as **Hiero**) is an approach to translation that builds on phrase-based translation by introducing the concept of hierarchical phrases. It was initially introduced by Chiang (2005) and expanded by Chiang (2007). Hiero served as an inspiration and a starting point for the graph-to-string translation and realization approaches described in this thesis. In this background section I therefore describe the hierarchical phrase-based grammar and how it is extracted. In a similar vein, I describe the hierarchical phrase-based translation decoder in Section 5.1.

Hierarchical translation in Hiero is based on a synchronous context-free grammar formalism. Namely, hierarchical phrase pairs are formally productions of a synchronous context-free grammar. **Synchronous context-free grammar** (SCFG) is a generalization of a context-free grammar (CFG) that generates pairs of token sequences instead of a single token sequence. SCFGs were originally developed for compilation of programming languages (Lewis and Stearns, 1968; Aho and Ullman, 1969), but have been found to be useful for applications in natural language processing, such as machine translation (Chiang and Knight, 2006). Hiero SCFG consists of rewrite rules with an aligned pair of right-hand sides:

$$X \rightarrow \langle \gamma, \alpha, \sim \rangle \tag{4.1}$$

where X is a non-terminal symbol, γ and α are source and target sequences of terminal and non-terminal tokens, and $\sim$ represents a one-to-one correspondence between non-terminal symbols in γ and α (shown as co-indexation). Two additional rules, called glue rules, allow Hiero to function similarly to the phrase-based translation model by combining hierarchical phrases sequentially, which improves its robustness:

$$\begin{aligned} S &\rightarrow \langle S_0 X_0, S_0 X_0 \rangle \\ S &\rightarrow \langle X_0, X_0 \rangle \end{aligned} \tag{4.2}$$

A single non-terminal type, X, is used in Hiero (in addition to the start symbol, S, used in glue rules) as opposed to using many types of non-terminals as in some syntactic CFG parsing approaches (e.g., *VP*, *NP*). Shallow-n grammars are an extension to Hiero which introduce additional (linguistically uninformed) non-terminal types into Hiero SCFG in order to reduce overgeneration and adapt the grammar complexity to specific language pairs (Iglesias et al., 2009; de Gispert et al., 2010).

The Hiero SCFG is an automatically extracted grammar, not based on any linguistic intuitions. Hiero rules are extracted from a word-aligned corpus of triples $(f, e, \sim)$, where f is the source sentence, e is the target sentence, and $\sim$ is the word alignment between them (word alignment was described in Section 3.3.3). Hiero rule extraction is a two step process. Firstly, initial phrase pairs consisting of terminals only are identified from a triple $(f, e, \sim)$ so that (1) they contain at least a pair of aligned tokens and that (2) no token in the initial phrase is aligned to a token not contained in the phrase. Secondly, rules are constructed from the initial phrase pairs and non-terminal phrase pairs, which are constructed by finding phrases that contain other phrases and replacing them with non-terminal symbols.

The above rule extraction procedure results in a large volume of extracted rules. The large volume of rules has the negative consequence of slowing down training and decoding. Additionally, it results in **spurious ambiguity**, which is the case when the grammar produces many alternative derivations which have the same target sides and feature vectors. Spurious ambiguity makes decoding more expensive due to the need to consider increased volume of alternative derivations and is detrimental to model tuning (described in Section 5.6) due to reduced number of distinct hypotheses. In order to avoid these problems, additional constraints are applied to the extracted rule set. These are:

1. A rule must contain at least one token on each side to ensure it is based on some lexical evidence.
2. A rule must not contain unaligned tokens at the edges of the phrase pairs.
3. The length of initial phrases is limited to 10 tokens on both sides.
4. Rules are limited to five tokens (including non-terminal symbols) on the source side.
5. Rules can contain at most two non-terminal symbols (X_0 and X_1).
6. Adjacent non-terminals on the source side are prohibited.

The hierarchical phrase-based decoder uses a log-linear model defined over derivations D to choose the best derivation (i.e., the best sequence of SCFG rules):

$$P(D) \quad \propto \quad P_{LM}(e)^{\lambda_{LM}} \times \prod_{(X \to \langle\gamma,\alpha\rangle) \in D} w(X \to \langle\gamma, \alpha\rangle) \tag{4.3}$$

where $P_{LM}(e)^{\lambda_{LM}}$ is the language model score, $(X \to \langle\gamma, \alpha\rangle)$ is a rule in derivation D, and $w(X \to (\gamma, \alpha))$ is a function that assigns a weight to a rule using rule features (but excluding the language model feature):

$$w(X \to \langle \gamma, \alpha \rangle) = \prod_{i \neq LM} f_i(X \to \langle \gamma, \alpha \rangle)^{\lambda_i} \tag{4.4}$$

where f_i is a rule feature function and λ_i is the corresponding rule feature weight.

The model uses the language model probability and rule features to make its prediction. The following rule features are used by Hiero: bidirectional phrase probabilities $P(\gamma \mid \alpha)$ and $P(\alpha \mid \gamma)$, lexical weights $P_w(\gamma \mid \alpha)$ and $P_w(\alpha \mid \gamma)$, rule penalty $\exp(-1)$ (so that a preference for longer or shorter derivations can be learned), and word penalty $\exp(-\#T(\alpha))$ (counting the number of terminal tokens, so that a preference for longer or shorter outputs can be learned). Additionally, there is a penalty feature for glue rules, $\exp(-1)$.

4.2 Rule extraction

The goal of rule extraction is to produce a Synchronous Context-Free Grammar (SCFG) consisting of rewrite rules of the form:

$$r = X \to \langle g, e, \sim \rangle \tag{4.5}$$

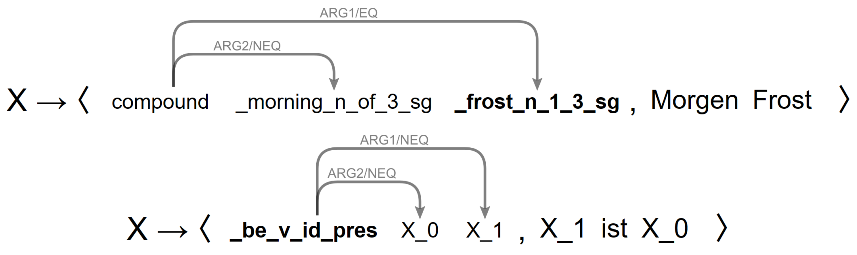
where r is either a terminal or non-terminal rule, X is the non-terminal left-hand side, g is a partial DMRS graph, e is a sequence of tokens, and $\sim$ is a one-to-one correspondence between non-terminal occurrences in g and e. The partial DMRS graph g, $g = (V, E)$, consists of nodes and edges. The *frontier* node v_f, $v_f \in V$, is the node which replaces a non-terminal node when the rule is applied hierarchically.

Examples of translation and realization terminal and non-terminal rules are shown in Figure 4.1. The one-to-one correspondence between source and target-side non-terminals, $\sim$, is shown implicitly via the non-terminal co-indexation between source and target sides, while the frontier nodes are shown in bold.

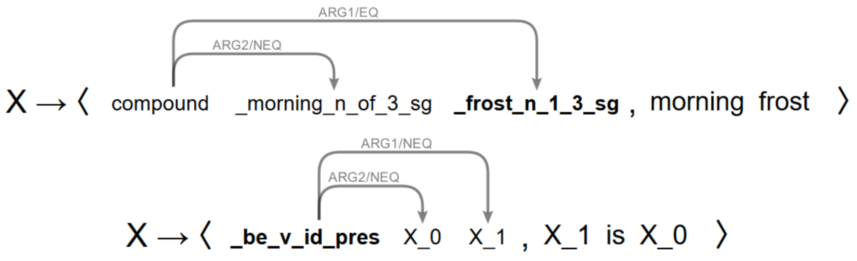
The rule extraction algorithm extracts rules such as the ones shown in Figure 4.1 from a sentence triple $\langle G, E, \sim \rangle$, where G is a DMRS graph, E is a token sequence, and $\sim$ is the alignment between them. A sentence triple $\langle G, E, \sim \rangle$ is the result of DMRS graph modelling and alignment, described in Sections 3.2 and 3.3 respectively. An example realization input sentence triple is shown in Figure 4.2. Observe that both rules shown in Figure 4.1b were extracted from the sentence input in Figure 4.2.

A graph-to-string SCFG derivation begins with a pair of non-terminals - a non-terminal node on the source side and a non-terminal symbol on the target side. In a single derivation step, a rule, such as the ones shown in Figure 4.1, is used to replace a non-terminal on both sides. On the source side a non-terminal node is replaced with the frontier node of the rule's partial DMRS graph. On the target side, the corresponding non-terminal symbol is replaced with the rule's target token sequence. An example derivation is shown in Figure 4.3. Since the graph-to-string SCFG is used to derive only the target-side strings in this thesis, the frontier nodes are not computed by the rule extraction algorithm and are omitted from subsequent example rules shown in this chapter.

The rule extraction algorithm extracts rules from a sentence triple by: (1) finding suitable (semantic) subgraphs of the input graph; (2) constructing terminal rules from those


ARG1/EQ
ARG2/NEQ
X → ⟨ compound _morning_n_of_3_sg _frost_n_1_3_sg , Morgen Frost ⟩
ARG1/NEQ
ARG2/NEQ
X → ⟨ _be_v_id_pres X_0 X_1 , X_1 ist X_0 ⟩

(a) Example of a terminal (top) and non-terminal (bottom) translation rule.


ARG1/EQ
ARG2/NEQ
X → ⟨ compound _morning_n_of_3_sg _frost_n_1_3_sg , morning frost ⟩
ARG1/NEQ
ARG2/NEQ
X → ⟨ _be_v_id_pres X_0 X_1 , X_1 is X_0 ⟩

(b) Example of a terminal (top) and a non-terminal (bottom) realization rule.

Figure 4.1: Examples of terminal and non-terminal translation and realization rules. In the remainder of the chapter I show only realization rules as examples.

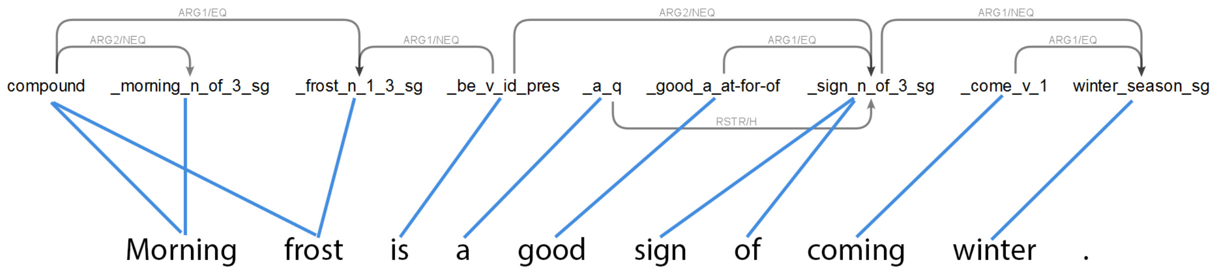


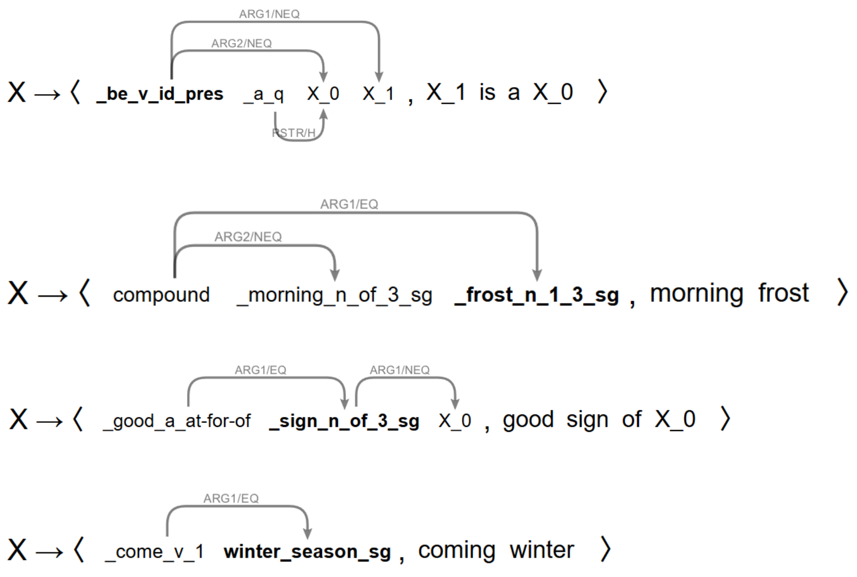
Figure 4.2: An example sentence triple $\langle G, E, \sim\rangle$ of DMRS graph, token sequence, and alignment between them, used as an input to the rule extraction algorithm.

subgraphs; (3) iteratively subtracting terminal rules from other rules (both terminal and non-terminal) to create non-terminal rules; and (4) filtering resulting rules.

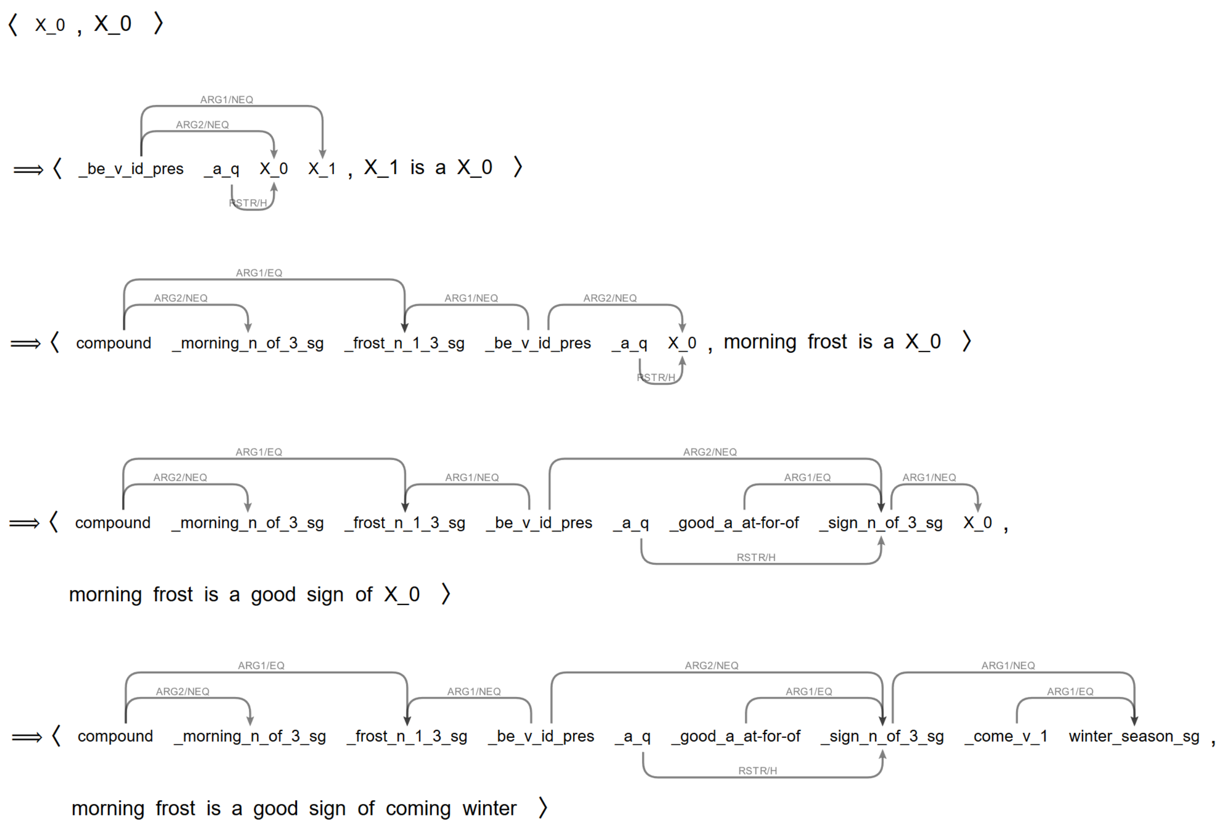
In the remainder of this section, I present the rule extraction algorithm in detail. As the algorithm is too complex to present in one go, I present sub-parts of the algorithm first. In Section 4.2.1 I describe the algorithm that creates a set of semantic subgraphs based on the semantic parse of the input graph. I continue by describing the algorithm for creating a non-terminal rule's source-side graph in Section 4.2.2. In Section 4.2.3 I present the constraints imposed on terminal and non-terminal rules. Finally, in Section 4.2.4, I give the full rule extraction algorithm and show the rules it extracts from the example in Figure 4.2.

4.2.1 Semantic subgraphs

Rule extraction algorithm begins by creating a set of subgraphs of the input DMRS graph. These subgraphs are consequently used to create rule source sides (g in Equation 4.5).


X → ⟨ _be_v_id_pres _a_q X_0 X_1 , X_1 is a X_0 ⟩
ARG1/NEQ
ARG2/NEQ
RSTR/H
X → ⟨ compound _morning_n_of_3_sg _frost_n_1_3_sg , morning frost ⟩
ARG1/EQ
ARG2/NEQ
X → ⟨ _good_a_at-for-of _sign_n_of_3_sg X_0 , good sign of X_0 ⟩
ARG1/EQ
ARG1/NEQ
X → ⟨ _come_v_1 winter_season_sg , coming winter ⟩
ARG1/EQ

(a) A set of rules in the order of their application for derivation in (b).


⟨ X_0 , X_0 ⟩
⟹ ⟨ _be_v_id_pres _a_q X_0 X_1 , X_1 is a X_0 ⟩
⟹ ⟨ compound _morning_n_of_3_sg _frost_n_1_3_sg _be_v_id_pres _a_q X_0 , morning frost is a X_0 ⟩
⟹ ⟨ compound _morning_n_of_3_sg _frost_n_1_3_sg _be_v_id_pres _a_q _good_a_at-for-of _sign_n_of_3_sg X_0 , morning frost is a good sign of X_0 ⟩
⟹ ⟨ compound _morning_n_of_3_sg _frost_n_1_3_sg _be_v_id_pres _a_q _good_a_at-for-of _sign_n_of_3_sg _come_v_1 winter_season_sg , morning frost is a good sign of coming winter ⟩
ARG1/NEQ
ARG2/NEQ
ARG1/EQ
RSTR/H

(b) Derivation of the example shown in Figure 4.2 using rules listed in (a).

Figure 4.3: Example derivation of a graph-to-string synchronous context-free grammar.

Not every subgraph of the input DMRS graph can be used as a rule source side, however. Namely, a subgraph needs to be created with respect to the semantic parse represented by the input DMRS graph. In this section I present an algorithm that traverses the DMRS graph with respect to its parse and creates a set of (what I will refer to as) semantic subgraphs.

In broad strokes, the algorithm creates a subgraph at a particular node n_{top} by traversing the input graph G, without crossing over to nodes that it has previously created subgraphs for. The algorithm is then recursively called on child nodes of n_{top}, repeating the procedure until a subgraph has been created at every node. In the case that a top node has a parent which has not yet been visited, a subgraph is first created for the parent node. The initial top node is `INDEX`, which points to the main verb of the sentence. When `INDEX` is not provided by the parser, `LTOP` is used instead (both `INDEX` and `LTOP` were introduced in Section 3.1.1). In the remainder of the section I present the full algorithm in two parts and demonstrate its operation on the example DMRS graph shown in Figure 4.2.

CreateSemanticSubgraphs$(G, n_{top}, \mathcal{V})$ is a recursive algorithm that creates induced subgraphs of graph G that correspond to the semantic parse represented by the graph. **Alg. 4.1** The algorithm starts at top node n_{top}. $\mathcal{V}$ is the set of previously visited nodes - an empty set when initially calling the algorithm.

1. (Initialize variables.) Initialize semantic subgraph set $\mathcal{G} \leftarrow \{\}$.
2. (Find unvisited parent nodes.) Set $\mathcal{P}_u \leftarrow P(n_{top}) \setminus \mathcal{V}$. Parent nodes of node n are nodes for which an edge originating in parent node and ending in n exists in G, $P(n) = \{p \mid p \in V(G) \text{ and } (p, n) \in E(G)\}$.
3. (Condition on number of unvisited parents.) If $|\mathcal{P}_u| == 0$:
    (a) (Create semantic subgraph.) Create semantic subgraph g, $g \leftarrow$ **CreateSemanticSubgraph**$(G, n_{top}, \mathcal{V})$ (using Algorithm 4.3 on page 69), and add it to $\mathcal{G}$.
    (b) (Find child nodes.) Find child nodes of node n. Child nodes of node n are nodes for which an edge originating in n and ending in a child node exists in G, $C(n) = \{c \mid c \in V(G) \text{ and } (n, c) \in E(G)\}$.
    (c) (Recurse with children.) For every child node c in $C(n)$, make a recursive call to **CreateSemanticSubgraphs**$(G, c, \mathcal{V} \cup \{n_{top}\})$, where the set of visited nodes $\mathcal{V}$ is extended with current top node n_{top}. $\mathcal{G}$ is extended with the resulting set of subgraphs from each recursive call.
4. (Condition on number of unvisited parents.) If $|\mathcal{P}_u| > 0$:
    (a) (Choose top parent.) Choose top parent p_{top} from $\mathcal{P}_u$ based on part of speech priority heuristic: conjunction > verb > preposition > quantifier > adjective > noun.
    (b) (Recurse with parent.) Call **CreateSemanticSubgraphs**$(G, p_{top}, \mathcal{V})$ and extend $\mathcal{G}$ with the the resulting set of subgraphs. Note that a subgraph for n_{top} is not created, but will eventually be created in one of the subsequent parent's recursive calls.
5. (Return.) Return $\mathcal{G}$.

```
be, {}
    → sign, {be}
        → a, {be}
            → sign, {be,a}
                → good, {be,a}
                    → sign, {be,a,good}
                        → winter, {be,a,good,sign}
                            → come, {be,a,good,sign}
                                → winter, {be,a,good,sign,come}
    → frost, {be}
        → compound, {be}
            → morning, {be,compound}
            → frost, {be,compound}
```

Figure 4.4: Trace of **CreateSemanticSubgraphs** algorithm for graph in Figure 4.2 and starting top node `be`. At each step, the current pair $(n_{\text{top}}, \mathcal{V})$ is shown. Arrows indicate recursive calls of the algorithm: green arrows for children and red arrows for parents. Creation of a subgraph from a node is indicated in bold. Note that a single subgraph is created at every node.

An algorithm trace for the example graph in Figure 4.2 is shown in Figure 4.4. The algorithm execution starts at the `INDEX` node `be` as n_{top} (I will refer to nodes by their lemmas for brevity) and an empty set of previously visited nodes $\mathcal{V}$. A subgraph is created from node `be`. The algorithm is then recursively called on both child nodes of `be`: `sign` and `frost`, which is indicated by nested green arrows. The set of previously visited nodes is expanded with node `be`.

However, a subgraph is not created for `sign` node in the first instance. This is because `sign` node has two parent nodes which have not yet been visited: `a` and `good`. Based on the parent priority heuristic (algorithm step 4a), `a` (a quantifier) is chosen as top parent over `good` (an adjective). The algorithm is then recursively called for node `a` without expanding set $\mathcal{V}$. After creating a subgraph at node `a`, a recursive call is made to its child, `sign`, again with expanded $\mathcal{V}$. However, as there is still an unvisited parent node (`good`), the process is repeated until $\mathcal{V} = \{\texttt{be}, \texttt{a}, \texttt{good}\}$, at which point a subgraph at `sign` node can finally be created. Algorithm execution then proceeds with the child nodes of `sign`.

Subgraph creation at a give node in Figure 4.4 is indicated with the node name shown in bold. A subgraph is created at a node by traversing as much of the DMRS graph as possible without traversing a previously visited node. The full algorithm is given below.

Alg. 4.2 **CreateSemanticSubgraph**$(G, n_{top}, \mathcal{V})$ is an algorithm that creates a semantic subgraph of graph G, starting at node n_{top}, and constrained by the set of previously visited nodes $\mathcal{V}$. An example of creating a semantic subgraph is shown in Figure 4.5.

1. (Initialize subgraph node set.) Initialize subgraph node set $\mathcal{N} = \{n_{top}\}$.
2. (Initialize traversing set.) Initialize set of nodes to traverse $\mathcal{T} = A(n_{top}) \backslash \mathcal{V}$. $A(n)$ defines adjacent nodes of node n, $A(n) = \{a \mid a \in V(G) \text{ and } ((a, n) \in E(G) \text{ or } (n, a) \in E(G))\}$. The set of previously visited nodes $\mathcal{V}$ is excluded from the traversing set.
3. (Repeat.) While the traversing set is not empty, $|\mathcal{T}| > 0$:

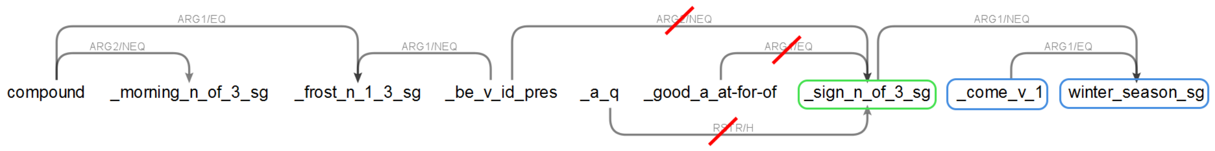


Figure 4.5: Semantic subgraph created for $n_{\text{top}} = \texttt{sign}$ and $\mathcal{V} = \{\texttt{be}, \texttt{a}, \texttt{good}\}$. The resulting subgraph nodes are shown with boxes around them, while the set of previously visited nodes is indicated by crossed edges leading to them.

   (a) (Select a node.) Select a random node from the traversing set, $n \in \mathcal{T}$.

   (b) (Extend subgraph node set.) Add node n to the subgraph node set, $\mathcal{N} = \mathcal{N} \cup \{n\}$.

   (c) (Extend traversing set.) Extend the traversing set with new adjacent nodes, $\mathcal{T} = \mathcal{T} \cup (A(n) \setminus \mathcal{V})$. The set of previously visited nodes $\mathcal{V}$ is again excluded from the traversing set.

4. (Create subgraph.) Create induced subgraph g from graph G with node set $V(g) = \mathcal{N}$.

5. (Return.) Return subgraph g.

The example in Figure 4.5 corresponds to line 6 of Figure 4.4: $n_{\text{top}} = \texttt{sign}$ and $\mathcal{V} = \{\texttt{be}, \texttt{a}, \texttt{good}\}$. The subgraph created without traversing nodes in $\mathcal{V}$ contains the following nodes: $\{\texttt{sign}, \texttt{come}, \texttt{winter}\}$. The full set of semantic subgraphs created for the input graph in Figure 4.2 is shown in Figure 4.6. Nine semantic subgraphs were created in total, at each of the nine node names shown in bold in Figure 4.4.

4.2.2 Graph subtraction

In the previous section I presented the algorithm for creating a set of terminal subgraphs of the input DMRS graph G corresponding to the semantic parse of the input graph. After terminal subgraphs are created, the rule extraction algorithm uses them as source sides to create terminal rules. The procedure for doing so is relatively straightforward and is presented as part of the rule extraction algorithm in Section 4.2.4.

The algorithm then proceeds to create non-terminal rules. It does so by comparing pairs of rules and checking whether one rule's source side is a subgraph of the other rule's source side (the same is done to check subsequence of target sides). When a subgraph relationship is found between a pair of source sides (and subsequence relationship exists between target sides), a non-terminal source side is created by subtracting the subgraph from the other graph and replacing it with a non-terminal node. In this section I describe the algorithm that achieves this. An example of graph subtraction is shown in Figure 4.7.

CreateNonterminalGraph(g, g_T, i) is an algorithm that given a graph g and its subgraph g_T replaces the occurrence of subgraph g_T in g with a non-terminal node X_i. **Alg. 4.3**

1. (Initialize non-terminal graph.) Initialize g_{NT} with g, $V(g_{NT}) = V(g)$ and $E(g_{NT}) = E(g)$.

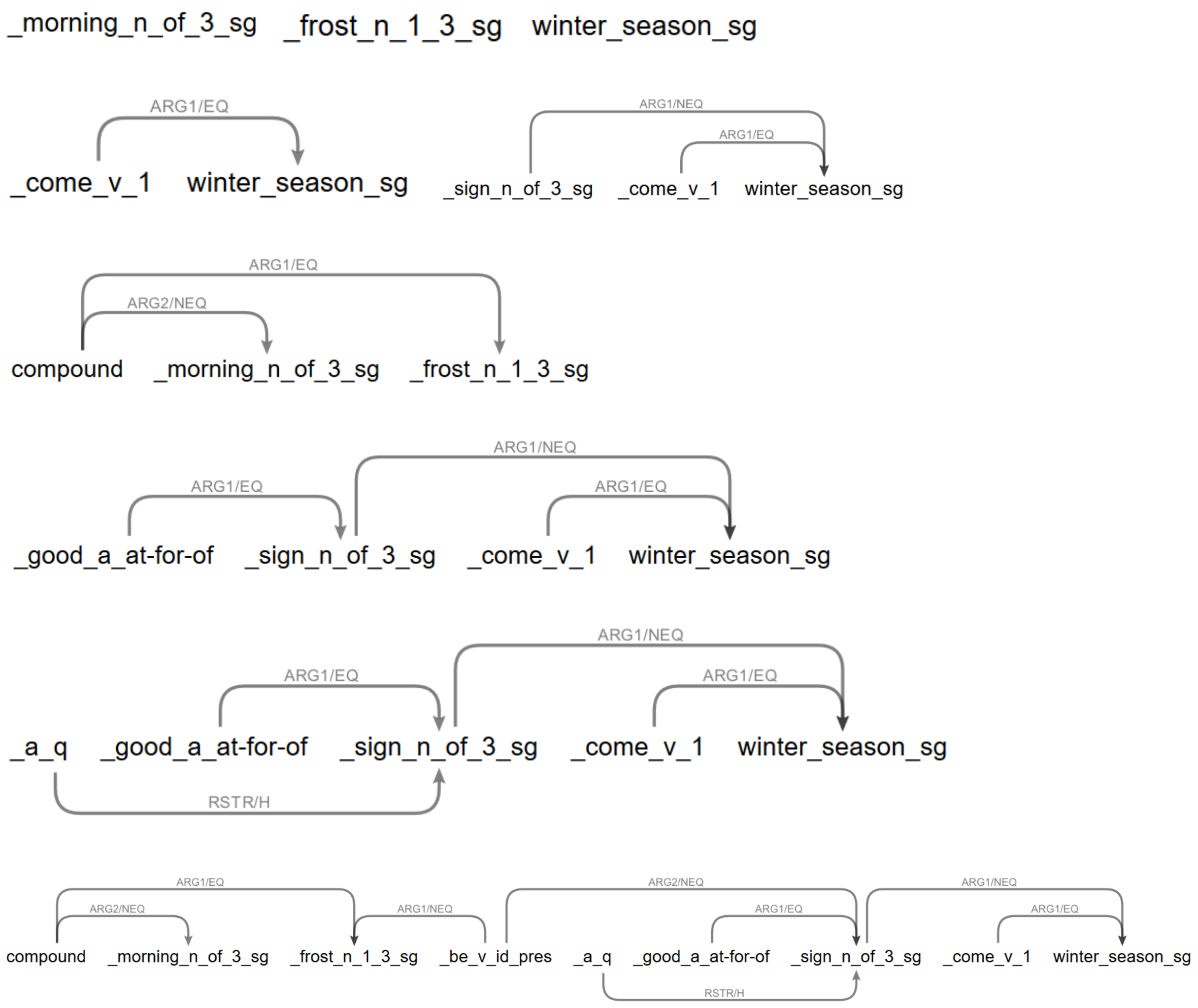


Figure 4.6: Nine semantic subgraphs constructed using the **CreateSemanticSubgraphs** algorithm for the initial graph shown in Figure 4.2.

2. (Remove intersected nodes.) Remove nodes of g_T from g_{NT}, $V(g_{NT}) = V(g_{NT}) \setminus V(g_T)$. Note that this leaves edges in g_{NT} without origin or end nodes.

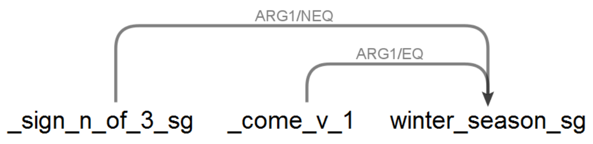
3. (Remove intersected edges.) Remove edges of g_T from g_{NT}, $E(g_{NT}) = E(g_{NT}) \setminus E(g_T)$.

4. (Remove incoming orphaned edges.) Remove edges with origin node in g_T and end node in g_{NT}, $E(g_{NT}) = E(g_{NT}) \setminus \{e \mid e = (n_o, n_e), n_o \in V(g_T), n_e \in V(g_{NT})\}$. This removes edges whose origin node was removed.

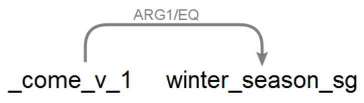
5. (Add nonterminal.) Add a non-terminal node: $V(g_{NT}) = V(g_{NT}) \cup \{X_i\}$.

6. (Replace outgoing orphaned edges.) Define the set of edges $\mathcal{E}_O$ as edges with origin node in g_{NT} and end node in g_T, $\mathcal{E}_O = \{e \mid e = (n_o, n_e), n_o \in V(g_{NT}), n_e \in V(g_T)\}$. Then each edge in $\mathcal{E}_O$ is replaced with an edge that has the same origin node as the original and has X_i as the end node: $f : e = (n_o, n_e) \longrightarrow e = (n_o, X_i)$. This replaces edges whose end node was removed.

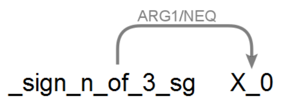
7. (Return.) Return non-terminal graph g_{NT}.



(a) Larger of the input graphs.



(b) Subgraph input graph.



(c) Non-terminal graph resulting from subtraction and replacement.

Figure 4.7: Graph subtraction example. The graph in (b) is subtracted from the graph in (a) and replaced with a non-terminal node `X_0` to produce the graph in (c).

Figures 4.7a and 4.7b show example input graph and its subgraph respectively, while Figure 4.7c shows the resulting graph after the algorithm replaces the subgraph with a non-terminal node.

4.2.3 Rule constraints

So far, I have described how semantic subgraphs are obtained from the input graph (Section 4.2.1) and how a non-terminal graph is created from a pair of graphs (Section 4.2.2). Before giving the full rule extraction algorithm in the following section (Section 4.2.4), I describe the constraints imposed on the extracted rules. The rule constraints extend the Hiero rule constraints described in Section 4.1 to graph-to-string translation. They ensure that extracted terminal and non-terminal rules (1) correspond to the relationship observed between input source graph and target string via the alignment between them, (2) are useful for translation and realization tasks, and (3) can be used tractably by the decoding algorithms described in Chapter 5. I start by defining the terminal and non-terminal constraints and continue with an informal explanation and counter-examples.

Given the input sentence triple $\langle G, E, \sim \rangle$, a **terminal rule** $r_T = X \rightarrow \langle g_T, e_T, \sim \rangle$ conforms to the following constraints:

1. g_T is a connected subgraph of G consisting of terminal nodes and e_T is a subsequence of E consisting of terminal tokens.

2. g_T node set consists of at least one node aligned to at least one token of e_T.

3. No node in $V(G) \setminus V(g_T)$ is aligned to a token in e_T and no token in $E = e_1 e_T e_2$ other than tokens in e_T is aligned to a node in g_T.

4. g_T is *semantically complete*. A graph g_T is semantically complete if for every node $n_i \in V(g_T)$ holds that $\{n_j \mid (n_i, n_j) \in E(G)\} \subseteq V(g_T)$.

Given the sentence triple $\langle G, E, \sim \rangle$, a **non-terminal rule** $r_T = X \rightarrow \langle g_{NT}, e_{NT}, \sim \rangle$ conforms to the following constraints:

1. g_{NT} is a connected graph consisting of terminal and non-terminal nodes and e_{NT} is a sequence of terminal and non-terminal tokens.
2. g_{NT} node set consists of at least one terminal node aligned to at least one terminal token of e_{NT}.
3. No node in $V(G) \setminus V(g_{NT})$ is aligned to a token in e_{NT} and no token in $E = e_1 e_{NT} e_2$ other than tokens in e_{NT} is aligned to a node in g_{NT}.
4. No edge originates from any non-terminal node of g_{NT}.
5. No non-terminal node of g_{NT} assumes underlying structure.

Since non-terminal rules are created by subtracting terminal rules from one another (as described in Section 4.2.2), the semantic completeness constraint imposed on terminal rules extends to non-terminal rules and is therefore not repeated there.

I demonstrate the purpose of terminal and non-terminal rule constraints by showing example rules breaking each of them in turn in Figures 4.8 and 4.9 respectively. Initially, Figure 4.8a shows a terminal rule extracted from the example in Figure 4.2 that conforms to all terminal rule constraints.

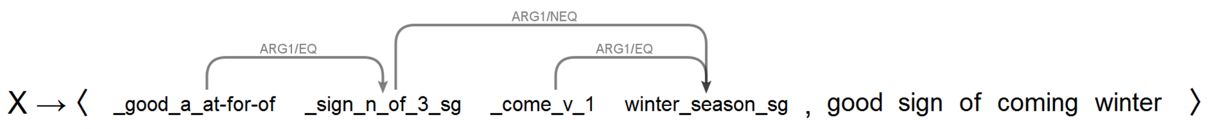
Terminal rule constraint 1 ensures that there exists a well-defined method of applying a rule to an input graph in translation or realization (as opposed to the indeterminate way a disconnected graph could potentially be applied to an input graph). Figure 4.8b shows an example of a rule with a disconnected graph on the source side, breaking the constraint.

Terminal rule constraint 2 prevents extracting insertion rules on one hand (empty source side) or deletion rules on the other hand (empty target side). The former are especially undesirable as they would introduce many contending rules at decoding time for every input graph without discrimination. The constraint ensures that there is some lexical evidence observed on the source side before a rule is applied. The deletion rules, on the other hand, have the potential to significantly increase the size of the grammar. Instead, deletion of source graph nodes is enabled by general deletion rules introduced in Section 6.1.2. Figure 4.8c shows an example of a rule with an empty target side, breaking the constraint.

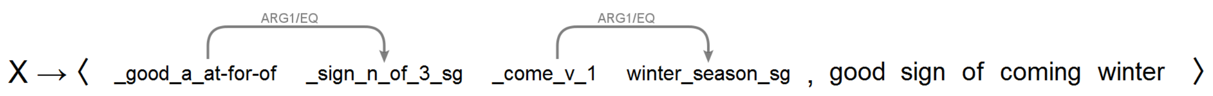
Terminal rule constraint 3 corresponds to the equivalent Hiero alignment constraint. It ensures that rules are extracted based on the evidence in the sentence pair and alignment between them. That is, no input graph node (or an input sentence token) that is aligned to a token (or a node) in a rule is missing from the rule. Figure 4.8d shows an example of a rule with missing alignments, breaking this constraint. Namely, tokens `sign of` aligned to the rule's source node `_sign_n_of_3_sg` are missing from the rule target side.

Finally, terminal rule constraint 4 ensures that the full argument structure of every predicate is extracted with the rule. Figure 4.8e shows an example of a semantically incomplete rule. Namely, the argument of node `_sign_n_of_3_sg` (connected with `ARG1/EQ` edge in the input graph, as shown in Figure 4.2) is missing from the example rule.

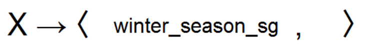
An example of a non-terminal rule conforming to all non-terminal constraints is initially shown in Figure 4.9a. Non-terminal constraints 1 and 3 are very similar in function to their terminal counterparts. Non-terminal constraint 1 differs from terminal constraint 1


ARG1/NEQ
ARG1/EQ
ARG1/EQ
X → ⟨ _good_a_at-for-of _sign_n_of_3_sg _come_v_1 winter_season_sg , good sign of coming winter ⟩

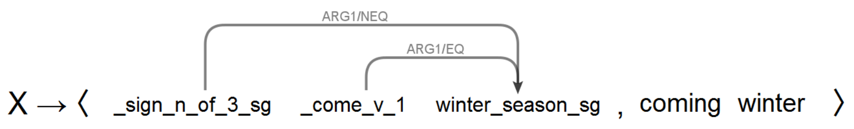
(a) Terminal rule conforming to all terminal rule constraints.


ARG1/EQ
ARG1/EQ
X → ⟨ _good_a_at-for-of _sign_n_of_3_sg _come_v_1 winter_season_sg , good sign of coming winter ⟩

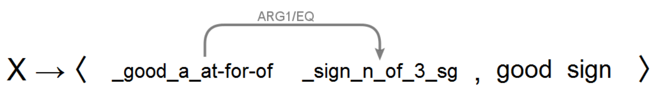
(b) Rule breaking terminal constraint 1 (disconnected subgraph).



(c) Rule breaking terminal constraint 2 (empty target side).

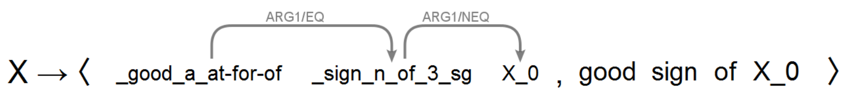

ARG1/NEQ
ARG1/EQ
X → ⟨ _sign_n_of_3_sg _come_v_1 winter_season_sg , coming winter ⟩

(d) Rule breaking terminal constraint 3 (misaligned).

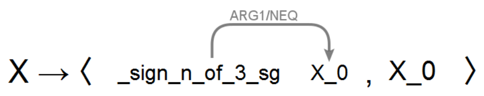


(e) Rule breaking terminal constraint 4 (semantically incomplete).

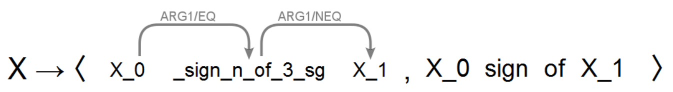
Figure 4.8: Examples of terminal rules conforming to terminal rule constraints (a) and not conforming to them (rest).


ARG1/EQ
ARG1/NEQ
X → ⟨ _good_a_at-for-of _sign_n_of_3_sg X_0 , good sign of X_0 ⟩

(a) Non-terminal rule conforming to all non-terminal rule constraints.



(b) Rule breaking non-terminal constraint 2 (non-lexicalised target side).



(c) Rule breaking non-terminal constraint 4 (non-terminal node with outgoing edge).

Figure 4.9: Examples of non-terminal rules conforming to non-terminal rule constraints (a) and not conforming to them (rest).

slightly, in that it does not require the source-side graph to be a subgraph and target-side sequence to be a subsequence of the original, due to the presence of non-terminal nodes and tokens. I consequently do not discuss them further.

Non-terminal constraint 2 is similar to its terminal counterpart (terminal constraint 2): it prevents extraction of insertion and deletion rules not grounded in lexical evidence. An example breaking the constraint is shown in Figure 4.9b - the example has a non-terminal token on its target side and no terminal tokens.

Non-terminal constraint 4 prevents extraction of rules that assume properties of non-terminal nodes. For example, non-terminal rule in Figure 4.9c assumes that node `X_0` has an argument of type `ARG1/EQ`. The constraint additionally prevents extraction of rules which would increase spurious ambiguity arising from arbitrary assignment of underlying nodes to two adjacent non-terminal nodes.

Finally, non-terminal rule constraint 5 relates to the way non-terminal graphs are constructed (algorithm described in Section 4.2.2). In the construction of g_{NT}, the g_T subgraph is replaced in g_{NT} by a non-terminal node. This constraint states that there is a single node in subgraph g_T to which nodes outside of g_T (but inside g_{NT}) connect to. That is, if there were multiple nodes in g_T to which nodes outside g_T would connect to, the non-terminal node which replaces g_T in g_{NT} would assume a certain underlying structure, which would result in extracted rules that are not context-free.

4.2.4 Rule extraction algorithm

In the previous subsections I introduced sub-parts of the rule extraction algorithm: in Section 4.2.1 I introduced the algorithm for extraction of semantic subgraphs from the input graph; in Section 4.2.2 I introduced the algorithm for subtracting a subgraph from a graph and replacing it with a non-terminal node; finally, in Section 4.2.3 I introduced terminal and non-terminal rule constraints imposed on extracted rules. In this section, I bring the sub-parts together to present the full rule extraction algorithm that extracts a set of terminal and non-terminal rules from a single sentence triple, consisting of a source graph, a target string, and the alignment between them.

Alg. 4.4 **RuleExtraction**$(G, E, \sim, N, V_{max})$ is an algorithm which, given a graph G, a target token sequence E and the alignment $\sim$ between them, extracts a set of rules $\mathcal{R}$. Parameter N controls the maximum number of non-terminals allowed in a rule; parameter V_{max} controls the maximum node set size of the source-side subgraph.

1. (Create candidate terminal subgraphs.) $\mathcal{G}_T \leftarrow$ **CreateSemanticSubgraphs**(G). Create terminal subgraphs of graph G with respect to the semantic parse of the graph.

2. (Create candidate terminal rules.) For every terminal subgraph g_T of $\mathcal{G}$, create a terminal rule $r_T = X \to \langle g_T, e_T, \sim\rangle$ by constructing e_T for subgraph g_T via alignments $\sim$.

3. (Apply terminal rule constraints.) For every candidate terminal rule r_T, check that it conforms to **terminal rule constraints**. Terminal rules that conform to the constraints are collected in $\mathcal{R}_T$.

4. (Initialize variables.) Initialize current working rule set $\mathcal{R}_{ws} \leftarrow \mathcal{R}_T$. Initialize result rule set $\mathcal{R}_{res} \leftarrow \mathcal{R}_T$.

5. (Repeat.) Repeat for N iterations to create rules with up to N non-terminals. Keep track of iteration number with $i \leftarrow 0$, incrementing it after each loop.

   (a) (Initialize new working rule set.) Set $\mathcal{R}_{new} \leftarrow \{\}$.

   (b) (Find candidate rule pairs.) For every rule $r = X \rightarrow \langle g, e, \sim\rangle \in \mathcal{R}_{ws}$, find terminal rules $r_T = X \rightarrow \langle g_T, e_T, \sim\rangle \in \mathcal{R}_T$, such that:

   - Terminal source-side graph g_T is a subgraph of graph g, $V(g_T) \subset V(g)$ and $E(g_T) \subset E(g)$. Determining the subgraph property is a trivial problem since both graphs are derived from the same initial graph G.
   - Terminal token sequence e_T is a subsequence of e.

   Each such pair of rules (r, r_T) is a candidate for creation of a non-terminal rule.

   (c) (Create candidate non-terminal rules.) For every candidate pair of rules (r, r_T), a non-terminal rule $r_{NT} = X \rightarrow \langle g_{NT}, e_{NT}, \sim\rangle$ is created by: (1) creating a non-terminal graph g_{NT} with **CreateNonterminalGraph**(g, g_T, i); and (2) creating a non-terminal token sequence e_{NT} by replacing subsequence e_T in e with a non-terminal token X_i.

   (d) (Apply non-terminal rule constraints.) For every candidate non-terminal rule r_{NT}, check that it conforms to **non-terminal rule constraints**. Non-terminal rules that conform to constraints are collected in $\mathcal{R}_{new}$.

   (e) (Update variables.) Extend result set $\mathcal{R}_{res}$ with $\mathcal{R}_{new}$. Set $\mathcal{R}_{ws} \leftarrow \mathcal{R}_{new}$. In the next iteration, the working rule set $\mathcal{R}_{ws}$ will contain newly created non-terminal rules with an additional non-terminal.

6. (Filter.) Filter result rule set $\mathcal{R}_{res}$ based on V_{max}.

7. (Return.) Return $\mathcal{R}_{res}$.

In order to keep the rule extraction procedure computationally tractable, two parameters can be used: (1) A rule contains at most N non-terminals; and (2) the size of the rule's source-side graph node set is limited to V_{max}. For all examples and experiments in the rest of the thesis, $N = 2, V_{max} = 5$. I remark on the effect of these constraints in Section 4.4.

The full set of rules extracted in Figure 4.2 is shown in Figure 4.10. In total, 20 rules were extracted: 6 terminal rules and 14 non-terminal rules (11 with one non-terminal, 3 with two non-terminals).

4.3 Grammar construction

In the previous section (Section 4.2) I introduced the rule extraction algorithm, which takes a single example triple (DMRS graph, target sentence, and their alignment) as input and extracts a set of rules. In order to construct a realization or translation grammar, rule extraction algorithm extracts sets of rules from a large training set of example triples (in most experiments in this thesis, training set sizes will contain millions of examples -

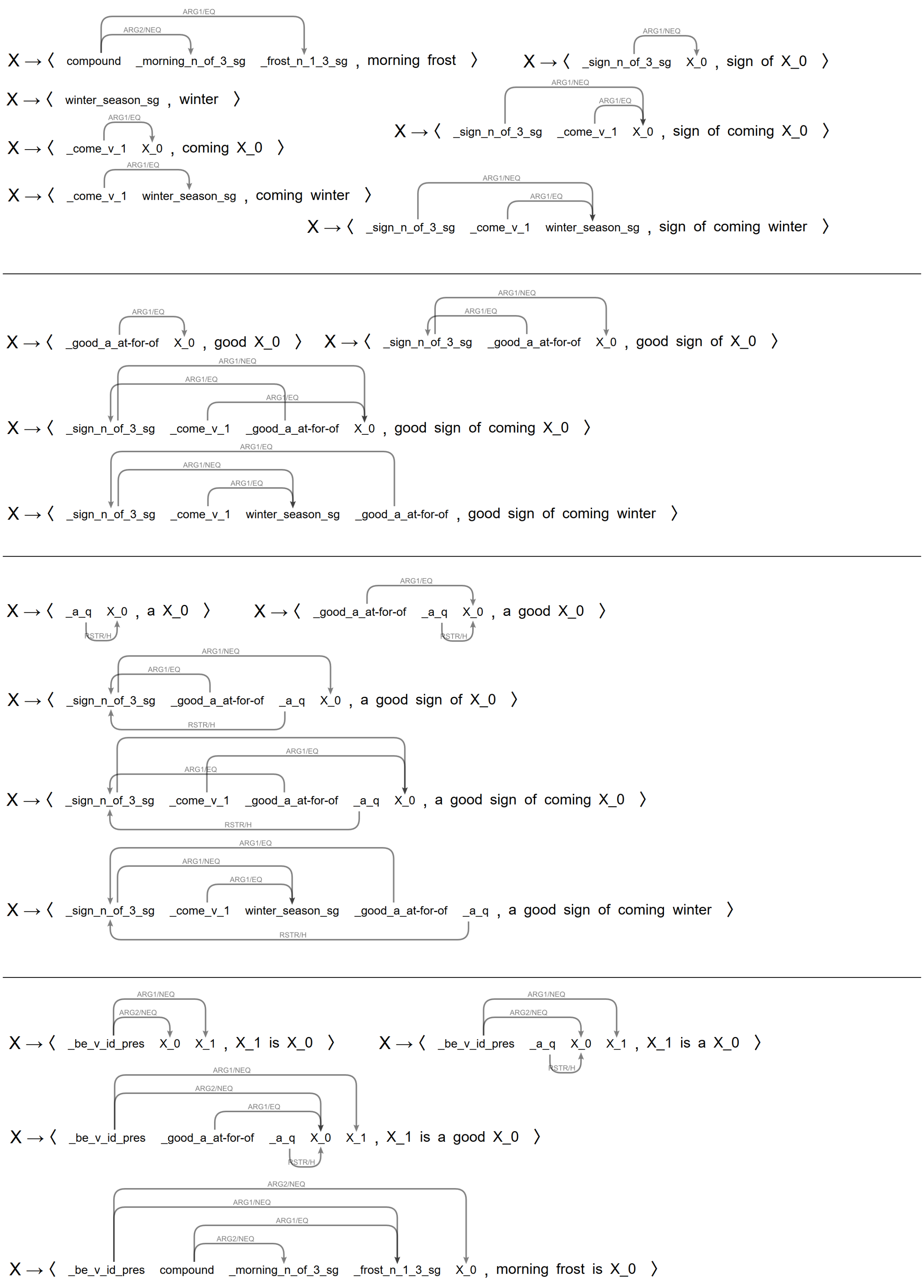

X → ⟨ compound _morning_n_of_3_sg _frost_n_1_3_sg , morning frost ⟩
ARG1/EQ
ARG2/NEQ
X → ⟨ _sign_n_of_3_sg X_0 , sign of X_0 ⟩
ARG1/NEQ
X → ⟨ winter_season_sg , winter ⟩
X → ⟨ _come_v_1 X_0 , coming X_0 ⟩
X → ⟨ _sign_n_of_3_sg _come_v_1 X_0 , sign of coming X_0 ⟩
X → ⟨ _come_v_1 winter_season_sg , coming winter ⟩
X → ⟨ _sign_n_of_3_sg _come_v_1 winter_season_sg , sign of coming winter ⟩
X → ⟨ _good_a_at-for-of X_0 , good X_0 ⟩
X → ⟨ _sign_n_of_3_sg _good_a_at-for-of X_0 , good sign of X_0 ⟩
X → ⟨ _sign_n_of_3_sg _come_v_1 _good_a_at-for-of X_0 , good sign of coming X_0 ⟩
X → ⟨ _sign_n_of_3_sg _come_v_1 winter_season_sg _good_a_at-for-of , good sign of coming winter ⟩
X → ⟨ _a_q X_0 , a X_0 ⟩
RSTR/H
X → ⟨ _good_a_at-for-of _a_q X_0 , a good X_0 ⟩
X → ⟨ _sign_n_of_3_sg _good_a_at-for-of _a_q X_0 , a good sign of X_0 ⟩
X → ⟨ _sign_n_of_3_sg _come_v_1 _good_a_at-for-of _a_q X_0 , a good sign of coming X_0 ⟩
X → ⟨ _sign_n_of_3_sg _come_v_1 winter_season_sg _good_a_at-for-of _a_q , a good sign of coming winter ⟩
X → ⟨ _be_v_id_pres X_0 X_1 , X_1 is X_0 ⟩
X → ⟨ _be_v_id_pres _a_q X_0 X_1 , X_1 is a X_0 ⟩
X → ⟨ _be_v_id_pres _good_a_at-for-of _a_q X_0 X_1 , X_1 is a good X_0 ⟩
X → ⟨ _be_v_id_pres compound _morning_n_of_3_sg _frost_n_1_3_sg X_0 , morning frost is X_0 ⟩

Figure 4.10: The full set of rules (20) extracted from the sentence triple shown in Figure 4.2. The rules are roughly grouped according to the extent of the input graph they were extracted from (i.e., graph coverage, introduced in Section 5.2) for readability.

see experiments in Chapters 6 and 7). Subsequently, the extracted sets of rules are used to construct a grammar. Grammar construction is the topic of this section.

Each rewrite rule can occur only once in the grammar. Therefore, in order to construct a grammar, extracted rules need to be aggregated. Aggregating rules requires determining whether two rules are the same, and consequently, whether two graphs are the same. This is known as the graph isomorphism problem. Graph canonization, described in Section 4.3.1, is a heuristic solution to the graph isomorphism problem. It transforms the rule's source-side graph, which is specific to the example triple from which the rule was extracted, into a canonical representation. Canonical representations of source-side graphs allow comparison and aggregation of rules extracted from any example triple. This reduces the problem of aggregating rules into a grammar to a comparison of strings solved in a distributed computation environment, as described in Section 1.5.

A grammar is used by the decoder to translate or realize an input graph. Graph canonization additionally enables grammar queries at decoding time (see Section 5.3). The result of such a query contains many competing rules that could be used to decode the input graph. The decoder uses rule features and a log-linear model in order to select the best set of rules to use (see Section 5.4.3). I conclude this section by describing grammar rule features and their computation in Section 4.3.2. The set of rule features consists of standard SMT features including bidirectional translation probabilities, rule count indicator features, and word and rule penalties. Additionally, I define rule type indicator features based on the DMRS structure in the rules' source-side graphs.

4.3.1 Graph canonization

In order for the rules extracted from individual sentences to be aggregated into a grammar, rules' source-side graphs need to be compared to each other so that their equality can be determined. Determining equality of two graphs is known as the graph isomorphism problem. The graph isomorphism problem belongs to the NP (nondeterministic polynomial time) complexity class, but is not known to belong to either P or NP-complete classes (Read and Corneil, 1977). Additionally, the rule application algorithm presented in Section 5.3 queries the grammar for rules that are applicable to a particular input graph. Querying the grammar also requires determining equality of a pair of graphs. In this section, I present a heuristic solution to the DMRS graph isomorpishm problem that addresses the related graph canonization problem in order to enable rule aggregation and grammar querying.

The rule extraction algorithm presented in Section 4.2 circumvents the graph isomorphism problem. The source-side graphs of rules created by the algorithm contain specific references to their input DMRS graph. Namely, the unique node ID information assigned by the parser is preserved from the DMRS graph to the rule source sides. Using node ID information simplifies and speeds up the rule extraction algorithm. For example, the subgraph property between a pair of graphs can be computed trivially by comparing their respective node IDs. However, node IDs are specific to an input example triple and do not generalize between different examples. Consequently, they cannot be used for rule aggregation discussed in this section. A solution is therefore needed to determine equality of two source-side graphs, which means addressing the graph isomorphism problem.

A related problem to graph isomorphism is graph canonization. Graph canonization is the problem of finding a graph's canonical form, such that the canonical forms of two isomorphic graphs are the same. Therefore, the graph isomorphism problem can be solved by transforming both input graphs to their canonical forms and comparing them (Read and Corneil, 1977). The graph canonization problem is at least as computationally hard as graph isomorphism (Read and Corneil, 1977). However, it presents a more convenient formulation of the graph isomorphism problem for rule aggregation and grammar querying. Namely, representing DMRS graphs as canonical strings makes it easy to compare millions of graphs in a large distributed computation environment (described in Section 1.5).

I propose a two-step heuristic solution to address the graph canonization problem. In the first step, the assignment of non-terminal indexes is made canonical. In the second step, a canonical ordering over the nodes of the graph is produced. The canonical ordering can be used to produce a canonical graph by augmenting each node label with its index in the canonical ordering. The canonical ordering can also be used to represent the graph in its canonical form as a string. I describe both steps in turn below.

The first step, the canonization of non-terminal index assignment, is only relevant to graphs with two or more non-terminals. In Algorithm 4.3 on page 69 the non-terminal node resulting from the subtraction of a subgraph is assigned index i, which is the subtraction loop iteration number. Therefore, the first subtracted subgraph will be replaced with node X_0, the second with X_1, etc. As there is no canonical order in which the subgraphs are subtracted from the graph, the assignment of non-terminal node indexes is arbitrary. In this step of the graph canonization solution, the non-terminal node indexes are reassigned so that they are the same between isomorphic graphs.

Alg. 4.5 **NonterminalIndexCanonization**(g) is an algorithm that given a graph g with two or more non-terminal nodes produces a graph with canonical non-terminal indexes.

1. (Create canonical non-terminal node representations.) Represent each non-terminal node $v_{NT} \in V(g)$ as a concatenation of string representations:

   (a) Lexicographically sorted list of incoming edges of node v_{NT},

   $$E^{-}(v_{NT}) = \{e \mid e = (v_i, v_{NT}), e \in E(g)\}$$

   Each incoming edge $e = (v_i, v_j)$ is represented as a concatenation of its origin node and edge label, (label(v_i), label(e)). The end node label is omitted since it is the non-terminal label with the index that is being reassigned.

   (b) Lexicographically sorted list of edges one-step removed from node v_{NT},

   $$E_{osr} = \{e_j \mid e_i = (v_i, v_{NT}), e_i \in E^{-}(v_{NT}), (e_j = (v_i, v_j) \text{ or } e_j = (v_j, v_i)), e_j \in E(g)\}$$

   Each one-step removed edge $e = (v_i, v_j)$ is represented as a concatenation of its origin node, end node, edge label, and relative edge direction, (label(v_i), label(v_j), label(e), direction(e, v_{NT})). Edge direction is relative to the non-terminal node v_{NT}, i.e. it has a value of 1 if it is an incoming edge to the node adjacent to v_{NT}, and a value of 0 if is an outgoing edge.

2. (Sort non-terminal nodes.) Sort the non-terminal node set by lexicographically comparing their canonical representations.

3. (Reassign non-terminal node indexes.) Reassign non-terminal node indexes based on their position in the sorted list of non-terminal nodes.

CanonicalNodeOrdering(g) is an algorithm that given a graph g produces a heuristic canonical ordering over its nodes $V(g)$. **Alg. 4.6**

1. (Create canonical node representations.) Represent each node v, $v \in V(g)$, as a concatenation of string representations:
   (a) Label of node v.
   (b) Lexicographically sorted list of incident edges of node v,
   $$E(v) = \{e \mid e = (v_i, v) \text{ or } e = (v, v_i), e \in E(g)\}$$
   Each incident edge $e = (v_i, v_j)$ is represented as a concatenation of its origin node, end node, edge label, and relative edge direction, (label(v_i), label(v_j), label(e), direction(e, v)).
   (c) Lexicographically sorted list of edges one-step removed from node v,
   $$E_{osr} = \{e_j \mid e_i = (v_i, v), e_i \in E(v), (e_j = (v_i, v_j) \text{ or } e_j = (v_j, v_i)), e_j \in E(g)\}$$
   Each one-step removed edge $e = (v_i, v_j)$ is represented as a concatenation of its origin node, end node, edge label, and relative edge direction, (label(v_i), label(v_j), label(e), direction(e, v)).
2. (Sort graph nodes.) Lexicographically sort node set $V(g)$ by their string representations to produce a node ordering.
3. (Return.) Return ordering of nodes in $V(g)$.

The output of the Algorithm 4.6 is an ordering over the nodes of the graph. From it, a canonical string representation of a graph can be created by concatenating node labels in their order. Take the source-side graph of the terminal rule in Figure 4.1 for example. The first step of the algorithm produces a string for each of the three nodes:[2]

```
(_morning, [compound-_morning-ARG2/NEQ-1], [compound-_frost-ARG1/EQ-0])
(compound, [compound-_frost-ARG1/EQ-0, compound-_morning-ARG2/NEQ-0], [])
(_frost, [compound-_frost-ARG1/EQ-1], [compound-_morning-ARG2/NEQ-0])
```

Each node representation is, as discussed above, composed of three parts: *(node label, lexicographically sorted list of incident edges, lexicographically sorted list of edges one-step removed)*. Each edge representation is, in turn, composed of four parts: *(origin node label, end node label, edge label, direction)*. Recall that edge direction is specified relative to the original node, i.e. it has a value of 1 if it is an incoming edge to the node adjacent to original node and a value of 0 if is an outgoing edge. Lexicographic sort (step 2 of the algorithm) over the three node representation strings yields the following canonical ordering of nodes:

[2]For brevity, only the lemma part of the node label is used in the example.

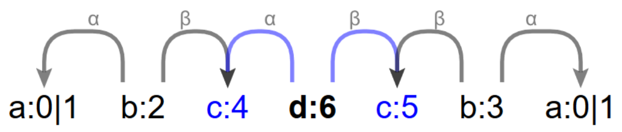


Figure 4.11: An example graph for which Algorithm 4.6 can produce two different node orderings. Node labels are `a`, `b`, etc., while the possible node orderings are indicated with indexes following the colon sign. 0|1 indicates that a pair of nodes could be arbitrarily assigned 0 and 1 between them.

```
[_frost, _morning, compound]
```

Finally, a canonical graph string representation can be created:

```
[_frost, _morning, compound], [2-0-ARG1/EQ, 2-1-ARG2/NEQ]
```

The above presented algorithm is a heuristic algorithm in order to keep the grammar construction procedure efficient. Consequently, the algorithm does not guarantee a correct solution for every input. Due to the heuristic of looking at adjacent and one-removed set of edges (but not further than that), the algorithms presented above can incorrectly determine two isomorphic graphs to be non-isomorphic.

An example graph is shown in Figure 4.11. Notice that in step 1 of the Algorithm 4.6, the same node representation string is created for both nodes with label `a`:

```
(a, b-a-α-1, b-c-β-0)
(a, b-a-α-1, b-c-β-0)
```

Since they share the same representation, the ordering between the two nodes with label `a` is not deterministic (either order could be produced). The two orderings over nodes of the graph produced by the algorithm in step 2 are indicated in Figure 4.11. From these two node orderings, two different graph representations can be created. Consequently, a pair of isomorphic graphs may happen to not share the same representation.[3]

The effect of encountering such a graph in practice would be that (1) a pair of extracted rules that are the same are not aggregated together and instead form two separate rules; and (2) the matching rules cannot be found in the grammar for a given query graph, produced in rule application (described in Section 5.3). However, neither scenario can happen in practice. The problematic graph requires a minimum of seven nodes in order to sufficiently distance the pair of offending nodes so that they share the same node representation. As stated in Section 4.2.4, the number of nodes in a source-side graph is limited to five in the current system.

[3]Notice that the graph in Figure 4.11 is not symmetric due to node `d`, which has two outgoing edges with two different labels, `0` and `1`. If the graph was symmetric, the two graph representations resulting from the two orderings would be the same, avoiding the negative consequences.

4.3.2 Rule features

We can observe several properties of rules after they have been aggregated. For instance, how many times has a rule been seen in the training set? How often was the source side translated with this rule's target side? How often was the target side a translation of this rule's source side? What is the structure of the source-side graph? These rule properties form rule features. Rule features are used in decoding by a log-linear model in order to score competing (realization or translation) hypotheses via the rules that were used to produce them (hypothesis scoring and selection is described in Section 5.4.3). Therefore, the purpose of rule features is to discriminate between competing rules in order to produce the best hypothesis given a grammar.

In this section, I introduce a set of dense rule features inspired by Bender et al. (2007) and Iglesias et al. (2009), and commonly used in statistical machine translation systems: bidirectional translation probabilities, rule count features, and word and rule penalties (I do not use the bidirectional lexical models). In addition to these commonly used features, I constructed a set of rule type features, which classify each rule into one of 14 types based on the structure of its source-side graph. I describe both sets of features below.

- **Source-to-target probability**, $P(e \mid g)$, is the probability of the target-side sequence e given the source-side graph g, estimated as a relative frequency over extracted rules (Koehn et al., 2003):

$$P(e \mid g) = \frac{\text{count}(e, g)}{\sum_e \text{count}(e, g)} \tag{4.6}$$

- **Target-to-source probability**, $P(g \mid e)$, is the probability of the source-side graph g given the target-side sequence e. Like source-to-target probability, it is estimated as a relative frequency over extracted rules:

$$P(g \mid e) = \frac{\text{count}(e, g)}{\sum_g \text{count}(e, g)} \tag{4.7}$$

- **Rule count indicator features**, namely 'rule count 1', 'rule count 2', and 'rule count more than 2' indicator features (an indicator feature equals -1 if a rule meets its condition, otherwise it is 0). These features are computed based on counts over extracted rules, for rule count 1:

$$c_1(e, g) = \begin{cases} 1 & \text{if count}(e, g) == 1 \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

  and similar for the other two features.

- **Word penalty**, $-\#T(e)$, is the negated count of the number of terminal tokens on the target side of a rule. The word penalty feature allows the log-linear model to learn a preference for shorter or longer outputs (Chiang, 2007).

- **Rule penalty**, -1, is a constant penalty for using a rule. The rule penalty feature allows the log-linear model to learn a preference for shorter or longer derivations (Chiang, 2007) (since competing hypotheses can use variable number of rules).

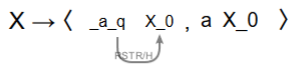
- **Rule type indicator features** is a collection of 14 features based on a rule's source-side graph. Each rule belongs to a single rule type.

Rule type indicator features take advantage of the DMRS structure in the source-side graphs to recognize different types and functions of rules (for example, basic versus non-basic rules, and noun compound versus verb phrase rules). I describe each rule type below,[4] but omit the details on how each type is detected.[5]

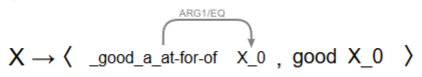
- **Basic noun** indicates (a source graph representing) a noun, a noun-like grammar predicate, or a nominalization relationship. Example rule:

  X → ⟨ winter_season_sg , winter ⟩

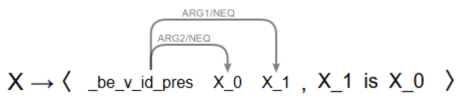
- **Basic quantifier** indicates a quantifier relationship (in the source-side graph) between a terminal quantifier node (its pos=q, see Table 3.1) and a non-terminal node. Example rule:



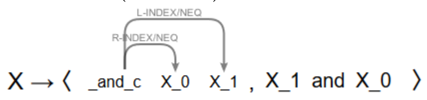
- **Basic modifier** indicates a modifier relationship between a terminal modifier node (pos=a) and a non-terminal node. Example rule:



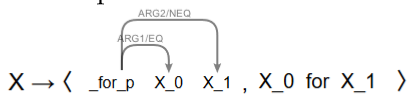
- **Basic verb** indicates a verb relationship between a terminal verb node (pos=v) and its non-terminal argument nodes. Example rule:



- **Basic conjunction** indicates a conjunction relationship between a terminal conjunction node (pos=c) and its non-terminal argument nodes. Example rule:



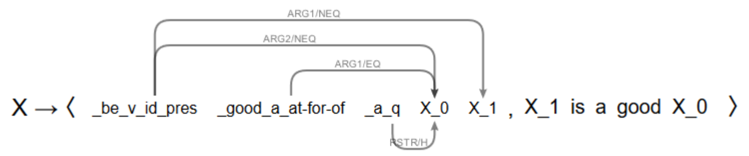
- **Basic preposition** indicates a preposition relationship between a terminal preposition node or preposition-like grammar predicate, and its non-terminal argument nodes. Example rule:



- **Verb phrase** indicates a graph representing a verb phrase. The top verb node must not have any incoming edges to ensure that it is not, for example, a conjunction phrase with a verb phrase as one of its arguments. Example rule:

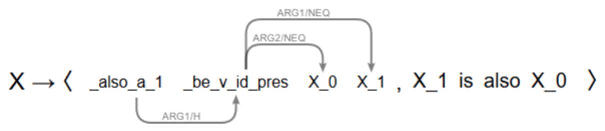
[4] I use syntactic terminology (e.g., verb phrase rule type) here for convenience. The terminology is usually interchangeable, but it is not guaranteed to be.

[5] In general, a rule is assigned the first rule type (in the order presented) matching the required set of conditions (indicated with its description).

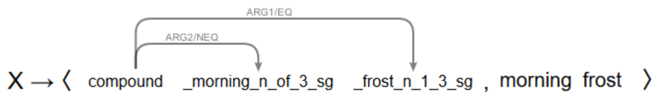

ARG1/NEQ
ARG2/NEQ
ARG1/EQ
X → ⟨ _be_v_id_pres _good_a_at-for-of _a_q X_0 X_1 , X_1 is a good X_0 ⟩
RSTR/H

Note the difference between verb phrase and basic verb rule types - while a basic verb rule captures only the immediate relationship between a verb and its arguments, a verb phrase rule contains additional nodes, capturing their realization/translation in the context of the verb node. Subsequent rule types share a similar relationship with their basic rule type counterparts.

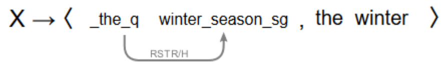
- **Modified verb phrase** indicates a verb phrase with a node (pos=a or grammar predicate `not`) modifying the verb. Example rule:



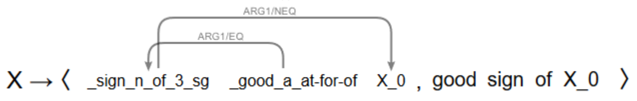
- **Noun compound** indicates a compound relationship between nouns. The compound grammar predicate node must not have any incoming edges (for the same reason as described above). Example rule:


ARG1/EQ
ARG2/NEQ
X → ⟨ compound _morning_n_of_3_sg _frost_n_1_3_sg , morning frost ⟩

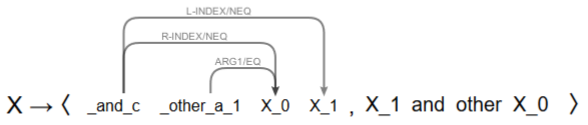
- **Full noun phrase** indicates a graph representing a noun phrase without any non-terminal nodes. Example rule:



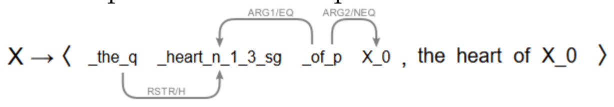
- **Partial noun phrase** indicates a graph representing a noun phrase which contains non-terminal nodes. Example rule:


ARG1/NEQ
ARG1/EQ
X → ⟨ _sign_n_of_3_sg _good_a_at-for-of X_0 , good sign of X_0 ⟩

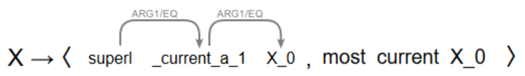
- **Conjunction phrase** indicates a graph representing a conjunction phrase. At least one node without any incoming edges must be a conjunction node (pos=c or selected grammar predicates). Example rule:



- **Preposition phrase** indicates a graph representing a preposition phrase. At least one node without any incoming edges must be a preposition node or a preposition-like grammar predicate. Example rule:



- **Other** is a special rule type that indicates that a source-side graph does not belong to any other defined rule type. Example rule:



In Section 4.4 I analyse the occurrence of rule types in the extracted translation and realization grammars.

One set of rule features not discussed here are non-grammar rule indicator features. They mark special purpose rules that are not extracted from the training set but are instead constructed for each decoding input graph individually. Non-grammar rule types include disconnected graph glue rules, deletion glue rules, CARG rules, and mapping rules. I introduce them in Section 6.1. Finally, language model score is an important feature used to score target-side output. However, the feature is not computed for a single rule, but is instead computed across the output of multiple rules (i.e., the hypothesis). I describe the details of how the language model score feature is computed in Section 5.4.3.

4.4 Rule extraction analysis

In this chapter I described the algorithms involved in extracting a grammar from a training set of examples. I conclude the chapter by analysing a number of translation and realization grammars in terms of their size, rule counts, and rule types (the latter two properties are referring to the grammar rule features described in Section 4.3.2). I start by analysing and comparing a translation and a realization grammar extracted using the full set of training examples. I continue by analysing and comparing grammars for both tasks across different training set sizes. I conclude with a brief discussion on the consequences of limiting the number of non-terminals and the number of nodes in a rule.

All grammars discussed in this section were estimated on the WMT15 English-German translation corpus. The corpus consists of 4.25 million training examples (a more detailed description is given in Section 6.2.1). For the translation task, the parallel sentence pairs are used in English to German direction, while the realization tasks uses only the English side. The English sentences of both tasks are parsed and the resulting graphs are processed (described in Section 3.2). The processed graphs are aligned to the corresponding German sentences in the case of translation, and to the original English sentences in the case of realization (described in Section 3.3). The result of these steps is a training set consisting of example triples (source-side graph, target-side sentence, and the alignment between them) required to extract a grammar.

The grammar sizes extracted from the full training set of 4.25 million examples are reported in Table 4.1. The realization grammar contains many more rules than the translation grammar (around 29 million versus 16 million rules, respectively), despite the fact that DMRS representations are closer to realization. This is a consequence of the alignments between source and target sides: (1) the translation alignments are obtained with a statistical word alignment algorithm and are therefore noisier than the realization alignments; (2) translation alignments contain more reordering, which can violate some of the rule constraints (described in Section 4.2.3), preventing rules from being extracted. In the following paragraphs, we investigate further differences between the grammars by comparing the composition of the two grammars in terms of rule counts and rule types.

In addition to the grammar sizes, the composition of the two grammars in terms of rule counts is also shown in Table 4.1. Rule count refers to the number of times a given

Rule count	Proportion of rules (%)		Number of rules (millions)	
	Translation	Realization	Translation	Realization
1	87.3	83.0	14.30	24.05
2	6.5	8.3	1.07	2.41
3-5	3.6	5.0	0.59	1.46
6-10	1.3	1.8	0.21	0.51
11-20	0.7	0.9	0.11	0.26
21-50	0.4	0.6	0.07	0.17
>50	0.3	0.4	0.05	0.12
All			**16.38**	**28.97**

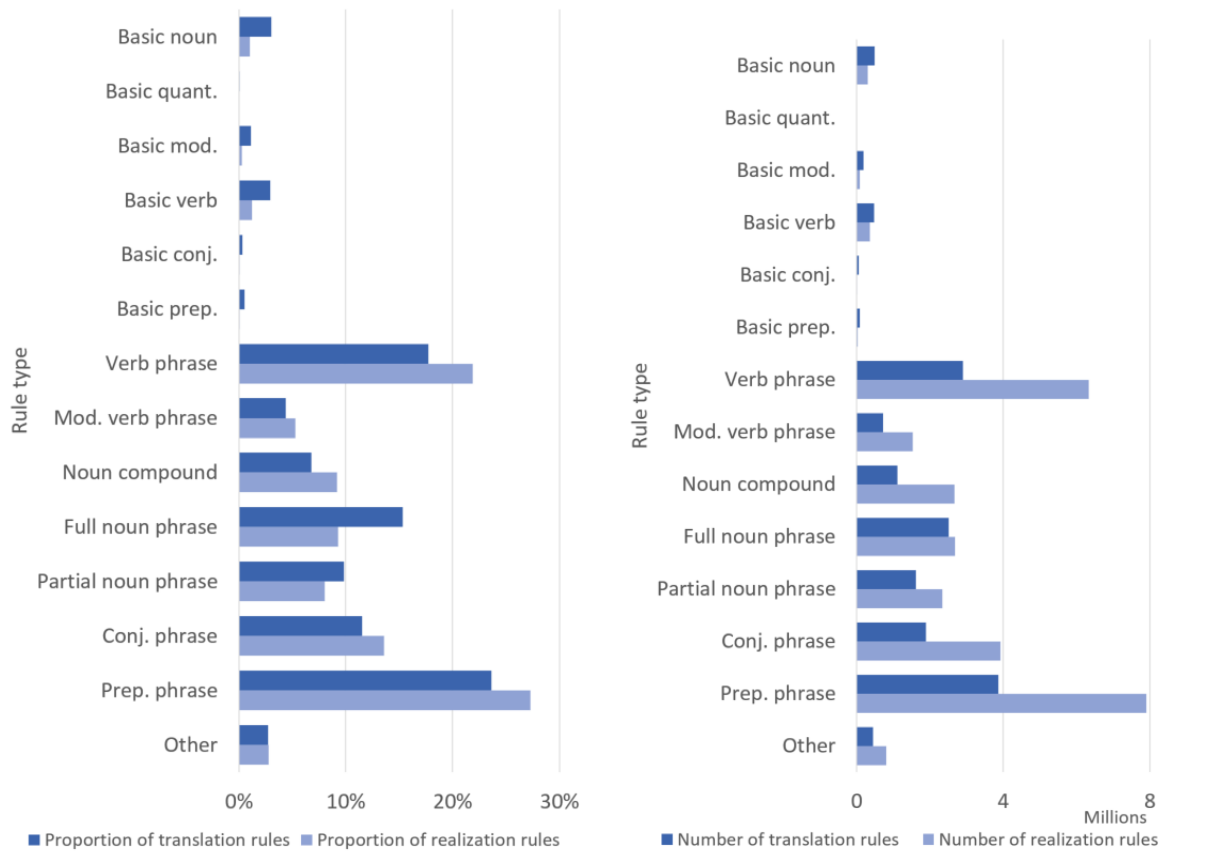
Table 4.1: Composition of the full translation and realization grammars in terms of rule counts (i.e., rules extracted once, twice, three to five times, etc.).

grammar rule has been extracted (recall that rule count is used as a feature, described in Section 4.3.2). The majority of both grammars consists of rules that have only been observed once (also referred to as singletons). There is a higher proportion of singletons in the translation grammar than in the realization grammar.

Another way of analysing the composition of the two grammars is according to the rule types defined in Section 4.3.2. They correspond to the differences in rule source-side graphs. For example, they can be divided into basic rule types (a single terminal node and potentially non-terminal nodes as its arguments) and non-basic rule types (containing additional terminal nodes as context); they can also be divided according to their function (e.g., decoding a verb versus decoding a noun). Every rule belongs to a single rule type. The distribution and the absolute number of rule types of the two grammars is shown in Figure 4.12.

Overall, preposition phrase is the most frequent rule type for both tasks, followed by verb phrase. This is expected because (1) both rule types occur frequently in the training examples (more so than, for example, conjunction phrase) and (2) they are the most inclusive rule types alongside conjunction phrase (for instance, a preposition phrase can contain a verb phrase or a noun phrase). Looking at the absolute numbers, the realization grammar has twice as many rules of preposition, verb phrase, and conjunction phrase types compared to the translation grammar. However, the translation grammar has a larger proportion of basic rule types compared to the realization grammar. Basic noun and basic verb rule types are especially prominent. Not only that, despite the translation grammar being significantly smaller than the realization grammar, it has a larger absolute number of all basic rule types.

Basic quantifier rules in particular account for a very small proportion of either grammar: 0.07% and 0.02% of the translation and realization grammars respectively. The translation grammar contains twice as many basic quantifier rules as the realization grammar (around 11 thousand versus 5 thousand). This is the case because there exist more ways of translating quantifiers between languages than there is ways of realizing them within a language (for example, English quantifier *the* can be translated into any of { *die*, *der*, *das*, *den*, *dem*, *des* ... } depending on the gender, case, and number, as well as additional contracted forms such as *am*). Additionally, translation rule extraction is affected by



(a) Proportion of rules of each type in the full translation and realization grammars.

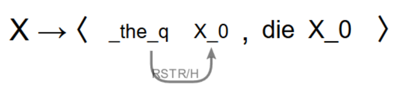
(b) Number of rules of each type in the full translation and realization grammars.

Figure 4.12: Composition of the full translation and realization grammars in terms of rule types, as defined in Section 4.3.2.

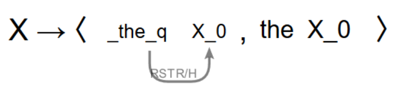
noisy alignments.

The eight most frequently extracted rules for both grammars are shown in Figure 4.13. Firstly, we can observe that the most frequent rules decode very common and general DMRS structures, namely quantifiers and common pronouns. Consequently, the top eight rules are all basic rules (with the exception of the realization grammar's preposition phrase rule) and there is a rough correspondence between the translation and realization rules. We can also observe that the translation grammar has five basic quantifier rules to the realization grammar's two, which reaffirms the evidence for the above explanation about basic quantifiers (conveniently, four rules translating node `_the_q` are in the top eight translation rules). Despite the top rules being general, their exact order will depend on the corpus from which they were extracted. If, for example, the corpus consisted of dialogue sentences, we would expect the top translation rules to include translation of `pron_2` to *du*, *Sie*, *dir*, *dich* etc.

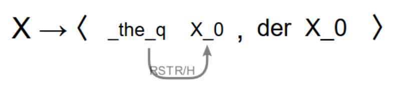
The analysis so far has focused on the two grammars extracted from the full training set. These grammars are subsequently used in translation and realization expirements reported in Chapters 6 and 7. I continue the analysis of rule extraction by investigating how the size of the training set affects the size of the grammar and its composition in terms of rule counts and rule types. I extracted four additional grammars for each of the translation and realization tasks by randomly sampling examples from the training set. A



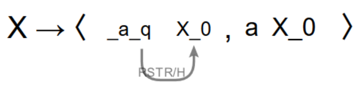
(a) $c = 572123$, Basic quant.



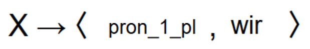
(b) $c = 2190855$, Basic quant.



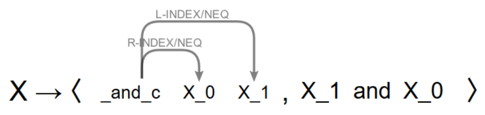
(c) $c = 392191$, Basic quant.



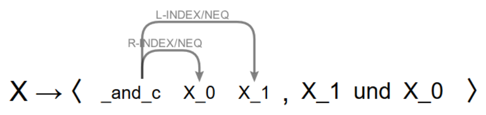
(d) $c = 848328$, Basic quant.



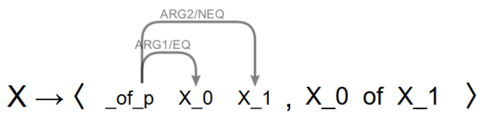
(e) $c = 307639$, Basic noun



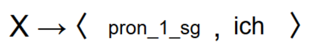
(f) $c = 652851$, Basic conj.



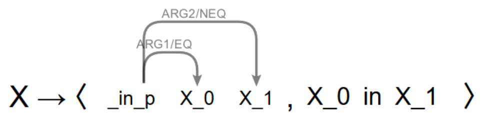
(g) $c = 235058$, Basic conj.



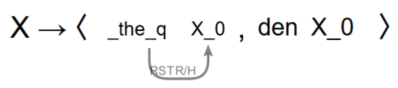
(h) $c = 647631$, Basic prep.



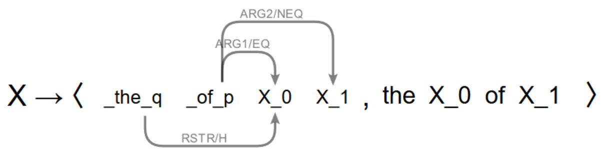
(i) $c = 231454$, Basic noun



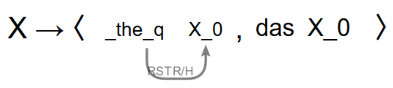
(j) $c = 589742$, Basic prep.



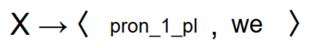
(k) $c = 189971$, Basic quant.



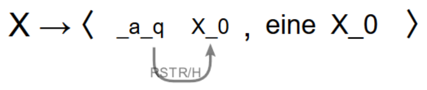
(l) $c = 431083$, Prep. phrase



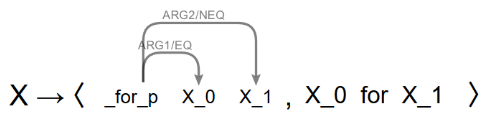
(m) $c = 150321$, Basic quant.



(n) $c = 429017$, Basic noun

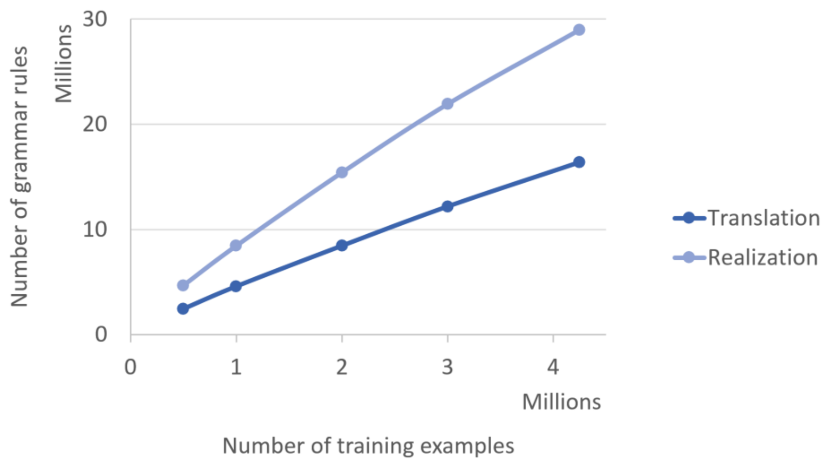


(o) $c = 142175$, Basic quant.

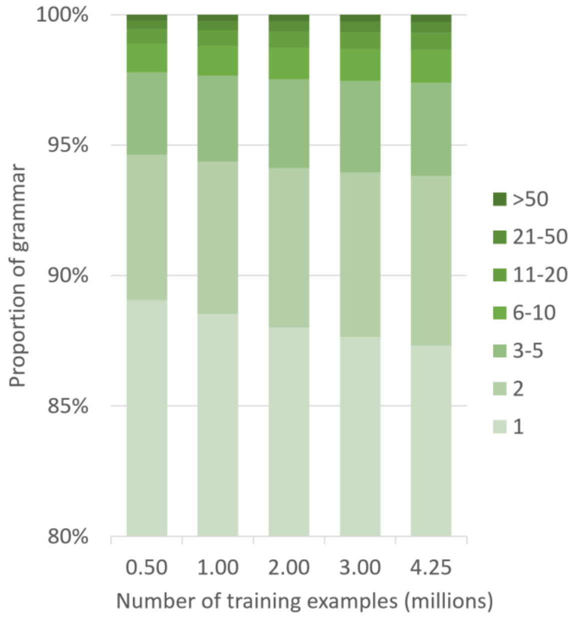


(p) $c = 376389$, Basic prep.

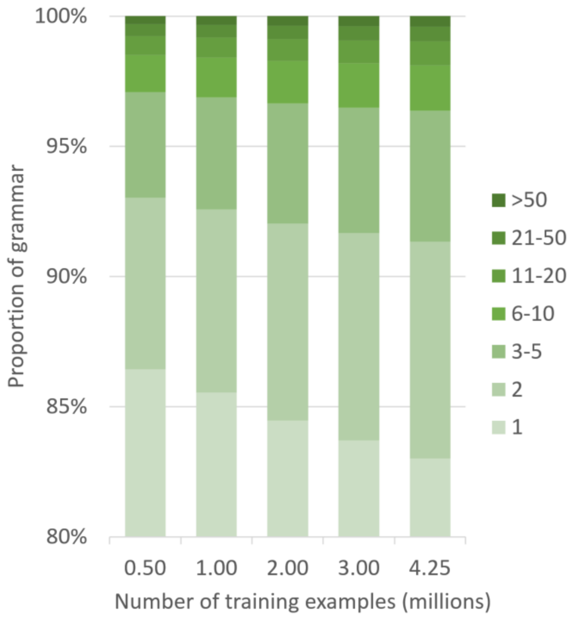
Figure 4.13: The top eight most frequently extracted rules in the full translation (left column) and realization (right column) grammars.



(a) Translation and realization grammar growth with increasing training set size.



(b) Translation grammar composition of rule counts.

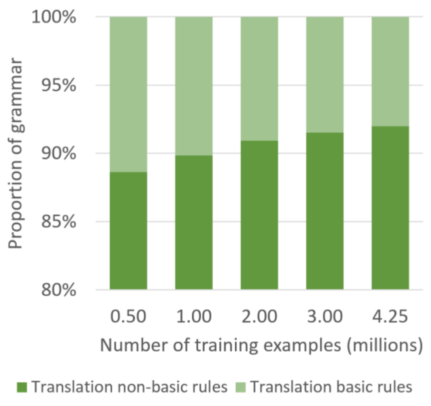


(c) Realization grammar composition of rule counts.

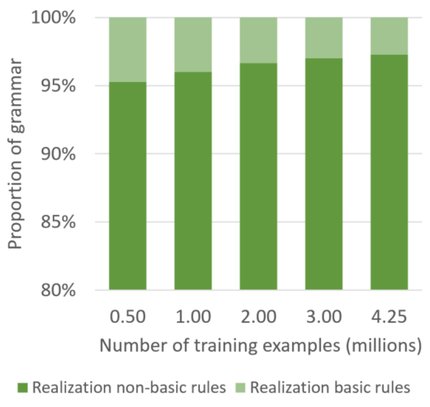
Figure 4.14: Translation and realization grammar size (top row) and grammar composition (bottom row) with increasing number of training examples. Grammar composition is shown in terms of proportion of rules with a given rule count (e.g., the proportion of rules observed more than 50 times).

total of five grammars for each task were trained on 0.5, 1, 2, 3, and 4.25 million training examples.

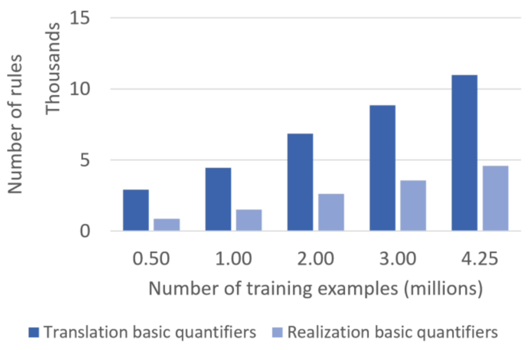
Figure 4.14a shows the grammar size at different training set sizes. We can observe that both the translation and the realization grammar size grows linearly with increasing training set size. The realization grammar growth is significantly steeper than the growth of the translation grammar. However, the factor of difference between them decreases with increasing training set size: the realization grammar is 1.91 times the size of the translation grammar at 0.5 million training examples, and only 1.76 times the size at 4.25 million training examples. This may indicate that the linear growth is tapering off



(a) Translation grammar composition of basic versus non-basic rule types.



(b) Realization grammar composition of basic versus non-basic rule types.



(c) Growth of the number of basic quantifier rules with increased training set size.

Figure 4.15: Translation and realization grammar rule type composition (top row) and number of basic quantifier rules (bottom row) with increasing number of training examples. Grammar rule type composition is shown in terms of basic versus non-basic rule types.

with the larger training set size. However, the full training set size is not large enough to determine that conclusively.

The grammar composition in terms of rule counts with increasing training set size is shown in Figures 4.14b and 4.14c for the translation and realization tasks respectively. The proportion of singletons decreases with increasing training set size for both tasks, but faster for the realization task. Conversely, the proportion of all other rule counts increases, with the proportion of rules extracted more than 50 times increasing the most (by 26% and 30% for the two tasks respectively).

Finally, we take a look at how the increase in training set size affects rule types in the grammar. I show the proportion of basic versus non-basic rule types with increasing training set size in Figures 4.15a and 4.15b for the translation and realization task respectively. As noted above with the full grammar sizes, we can observe that basic rule types form a larger part of the translation grammar compared to the realization grammar, at all training set sizes. However, the proportion of basic rule types decreases with increased training set size for both tasks. Intuitively, this makes sense, since basic rule types are more constrained compared to non-basic rule types (see description in Section 4.3.2).

Earlier in the section, I discussed the basic quantifier rule type in the full grammar. To complement that discussion, I show the number of rules with basic quantifier rule type as a function of increasing training set size in Figure 4.15c. We can observe that the translation grammar contains significantly larger number of basic quantifier rules at every training set size, reaffirming the discussion above. What may come as a surprise is that the number of basic quantifier rules keeps growing with additional training data, despite quantifiers being a closed class. This is because different translations/realizations of quantifiers exist, including non-literal translations. So more data encountered means that more possible ways of translating/realizing them are extracted. To some extent, this will also be due to the noise in alignments, in which case the extracted rules are incorrect.

The rule extraction algorithm presented in Section 4.2.4 requires specifying two parameters, the maximum number of non-terminals and the maximum number of nodes in a rule. Limiting the number of non-terminals and the number of nodes in a rule is necessary in order to prevent the grammar size increasing beyond what can be feasibly used by the decoder. All examples and experiments reported in this thesis limit the number of non-terminals to two and the number of nodes to five. These heuristic constraints match the Hiero constraints described in Section 4.1, where rules can contain at most two non-terminal symbols, and the source side can contain at most five tokens.

Limiting the number of nodes in a rule to at most five limits the size of the semantic context captured by a rule. However, since the semantic context in a DMRS graph is immediately adjacent, it is a less restricting limit compared to Hiero, where semantically related words can be spread more than five tokens apart. Limiting the number of non-terminals in a rule to two is more constraining. It means that no basic rule can be extracted for predicate nodes with three arguments or more. However, such predicate nodes are uncommon and the grammar will contain rules with partially specified arguments.

In summary, I found that the realization grammar is significantly larger that the translation grammar at all training set sizes, despite the DMRS representations being closer to the target sentences in realization than in translation. This is a consequence of the noise in translation alignments and translation reordering constraining rule extraction. The size of both grammars grows linearly with increasing training set size, but the growth of the realization grammar is significantly steeper than the growth of the translation grammar. In terms of grammar composition, the majority of rules of both grammars have only been observed once, but their proportion decreases with increasing training set size. Verb and preposition phrases are most frequent rule types encountered in both grammars, and, as anticipated, the proportion of basic rules decreases with increasing training set size.

4.5 Realization grammar comparison

The grammar presented in this chapter, henceforth referred to as HSSR (hierarchical statistical semantic realization) grammar, and the ERG are both used for realization of DMRS graphs (see Chapter 7). I conclude this chapter with a discussion of differences and similarities between them.

In the realization task the HSSR grammar encodes the relationship between DMRS graphs and strings. Similarly, the ERG can be seen as both defining the set of grammatical strings and providing a mapping between these strings and structures.[6] Unlike the ERG, the HSSR grammar does not constrain the set of grammatical strings or well-formed structures. This formal difference between the grammars is not a limitation of the HSSR

[6]The semantics is part of this structure and the DMRS used in this thesis is derived from it. Copestake (2002, p. 83) discuss this in the context of the formalism used by the ERG.

grammar, but is instead what makes the HSSR grammar robust, as demonstrated in Chapter 7.

The ERG is an instance of a typed feature structure grammar. The typed feature structure formalism is Turing equivalent, and is therefore capable of encoding anything computable. However, the ERG does not make full use of the capacity of the formalism. Similarly, the HSSR grammar does not make full use of the SCFG formalism (e.g., constraints in Section 4.2.3). Consequently, the following discussion is in terms of some phenomena that are captured in the ERG as opposed to HSSR, not the theoretical power of the formalisms they use.

- Since the ERG is a precision grammar, it is designed to constrain the set of grammatical strings to those which the grammarian considers well-formed.
- ERG constraints lead to a notion of global well-formedness. For instance, a string which consists of well-formed phrases will not be grammatical unless the phrases are related by rules and the full structure meets the root condition. On the other hand, HSSR grammar has no guarantees of well-formedness or grammaticality.
- HSSR rules are always lexicalised and do not incorporate the rich notion of features that the ERG uses. Consequently, ERG rules capture generalizations which HSSR rules do not. For example, HSSR rules will only capture agreement and subcategorization properties in some specific contexts rather than completely.
- The restriction to two non-terminals in HSSR was discussed in section 4.4. Although the ERG generally limits rules to be at most binary branching, the availability of the rich feature structures means that this restriction is not problematic (e.g., for ditransitives and other cases where a predicate has three arguments). Comparable constraints are not fully captured in HSSR.

Limiting our discussion to realization from well-formed DMRS, the main requirements of the realization system are (1) appropriate determination of lexemes from DMRS predicate symbols, (2) appropriate insertion of semantically empty lexemes, (3) correct morphophonology, and (4) ordering. Consequently, some limitations of the HSSR grammar compared to the ERG constraint system are irrelevant. For example, the fact that a verb such as ‘allow’ (in the sense of ‘allows (us) to see’) requires two complements will not be captured in the HSSR grammar. However, a well-formed DMRS will specify both complements, making the limitation of the HSSR grammar irrelevant. Similarly, the language model will capture some constraints that the HSSR rules do not (e.g., the phrase ‘I see’ will be preferred to ‘I sees’ when constructed using two (context-free) rules).

In this section, the discussion of differences and similarities between HSSR grammar/rules and ERG focused on the realization task. In the translation task, HSST (hierarchical statistical semantic translation) grammar is comparable to the combination of transfer rules and realization grammar in the DELPH-IN approach to machine translation (e.g., Bond et al. (2011), see Sections 2.4 and 7.4.3). Although the comparison is complicated by the transfer rules (in addition to the fact that the target grammar would be German), most of the points above still apply. However, in translation the input to the realization system is not necessarily a well-formed DMRS and realization acts as a filter on the outputs of transfer.[7]

[7]In Bond et al. (2011), ERG realization is used as a filter of ill-formed transferred MRS (personal communication with Francis Bond).

Chapter 5

Decoding

Decoding in statistical machine translation refers to the transformation of a sentence in the source language into a sentence in the target language. In hierarchical phrase-based translation (Hiero; Chiang, 2005, 2007), the input to the decoder is a sentence represented as a sequence of tokens. The Hiero decoder uses a synchronous-context free grammar as a resource from which translation rules are drawn (see Chapter 4). It typically organizes the translation rules into a CYK grid according to the part of the input sentence they translate. The decoder then proceeds to explore the CYK grid in order to construct a set of translation hypotheses. I describe Hiero decoding in more detail in the background Section 5.1.

Instead of a token sequence, the input to a decoder for the approach discussed in this thesis is a DMRS graph. This difference necessitates building a new type of a decoder that is able to transform the structure of a graph into a sequence of target tokens. Whereas token spans are used in a Hiero decoder to keep track of the parts of the input sentence covered by a rule, they cannot be used to keep track of the parts of a graph. Because of this, I introduce graph coverage in place of a token span, which can be represented as a bit vector with ones corresponding to the covered nodes. I describe graph coverage in more detail in Section 5.2.

Unlike in Hiero, where potentially many competing parses of an input sentence exist, the DMRS graph itself already encodes a preferred reading. Consequently, I propose a rule application algorithm in place of CYK parsing used by a Hiero decoder. The rule application in many ways mimics the rule extraction algorithm presented in Section 4.2. Given an input DMRS graph and a graph-to-string SCFG, the rule application algorithm constructs a graph coverage grid. The grid consists of cells containing graph-to-string rules, which can be used to transform the corresponding part of the graph into a sequence of tokens. The rule application algorithm starts by finding semantic subgraphs of the input DMRS graph, in the same way as the rule extraction algorithm. Likewise, non-terminal subgraphs are created by subtracting terminal subgraphs from other subgraphs. The terminal and non-terminal subgraphs are subsequently used to query the SCFG to obtain the set of all graph-to-string rules that could be applied to the input graph. The rule application algorithm is described in detail in Section 5.3.

The resulting graph coverage grid needs to be explored in order to construct a set of translation or realization hypotheses. In order to achieve this, I adapt the ideas behind HiFST (Blackwood et al., 2016), a Hiero decoder which compactly encodes the target-side hypothesis space as Finite State Transducers (FSTs), for graph-to-string decoding.

The resulting graph-to-string decoder encodes each graph coverage cell as a Recursive Transition Network (RTN), with rule non-terminals as pointers to cells lower in the grid. The full hypothesis space FST is constructed by recursively expanding the top RTN cell. A log-linear model is defined over the derivations and is used to compute the weight of the FST arcs. Finding the best hypothesis then corresponds to finding the shortest path in the hypothesis space.[1] I describe the hypothesis space construction and the best hypothesis search in Section 5.4.

The size of the hypothesis space grows with the size of the input graph and the size of the grammar. Consequently, decoding a large input graph using a large grammar can result in a high memory cost and a long decoding time. I introduce two strategies to alleviate this problem: (1) cell selection, which reduces the number of RTNs that need to be created and expanded without negatively impacting output quality, and (2) local pruning, an adaptation of local pruning used in HiFST, which reduces the size of cell RTNs via lossy pruning, allowing a trade-off between output quality and decoding efficiency. I describe the two strategies in Section 5.5.

The log-linear model used to score the hypotheses is based on rule features described in Section 4.3.2 and a language model. In order to tune the model parameter weights, I use the Minimum Error Rate Training (MERT; Och, 2003) procedure commonly used to tune statistical machine translation systems. In order to use MERT, I extend the hypothesis space construction to additionally keep track of derivations, following the ideas introduced by de Gispert et al. (2010). I describe model parameter tuning in Section 5.6.

As with rule extraction and grammar construction approaches, the proposed rule application algorithm and the graph-to-string decoder are task-agnostic: they can be applied to both translation and realization tasks without any modification. Consequently, contrasting the rule application algorithm and graph-to-string decoder performance on the two tasks makes for an informative comparison. I compare the graph-to-string decoder performance in terms of algorithm running time conditioned on the input graph size. In addition to running time, I compare the rule application algorithm performance between the two tasks in terms of the resulting number of applied rules. I report the results in Section 5.7.

5.1 Hierarchical phrase-based decoding

In this background section, I describe decoding in hierarchical phrase-based translation. I also describe HiFST, an approach to hierarchical phrase-based decoding using Finite State Transducers that simplifies and improves upon the cube pruning approach of Chiang (2007). The HiFST decoder, introduced by Iglesias et al. (2009) and improved by de Gispert et al. (2010), forms the basis of our graph-to-string decoder described in subsequent sections.

In Section 4.1 I described Hiero rule extraction procedure and the resulting synchronous context-free grammar. The approach defines a log-linear model over derivations, using rule features and a language model feature. The task of a Hiero decoder is then to find the highest-scoring translation. As noted by Chiang (2007), the decoding procedure would be

[1] An initial version of the rule application algorithm and the graph-to-string decoder was described in Horvat et al. (2015).

a straightforward dynamic-programming algorithm with a weighted context-free grammar if not for the language model. The reason for the complication arises because the language model features are non-local, that is it cannot be computed for a single rule. Instead, a language model score needs to be computed in the context of nested rules. That is to say, computing a language model score for `from X`$_0$ is not possible, whereas for `from London` is.

Hiero decoder uses a variant of the CYK (Cocke–Younger–Kasami) parser, CYK+ (Chappelier and Rajman, 1998), operating on the weighted context-free grammar. Informally, it populates the CYK grid (such as the one shown in Figure 5.1a) from the bottom-up by considering the input token subsequences and checking whether grammar rules joining them exist, while keeping track of model probabilities. It concludes when it reaches the top cell covering the entire input token sequence.

When constructing the CYK grid, the Hiero decoder uses beam search to manage the size of the search space. This means a maximum of β top candidates are kept in every CYK grid cell, potentially resulting in search errors. To improve decoding efficiency the Hiero decoder also employs hypothesis recombination, which merges equivalent items in a CYK cell, keeping the weight of the better scoring item.

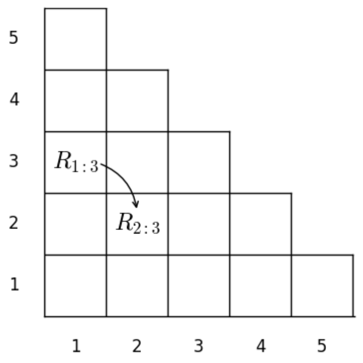
Adding a language model score to the Hiero decoding procedure described so far is achieved using cube pruning, which was introduced by Chiang (2007). Iglesias et al. (2009) introduced decoding with Weighted Finite State Transducers (WFST, see Section 5.4.1) as an alternative to cube pruning which is easier to implement and reduces the number of search errors significantly. They named the improved system **HiFST**.

HiFST uses a similar parsing procedure to parse the source sentence and determine the placement of grammar rules over the sentence's CYK grid. However, instead of using beam search and cube pruning, HiFST constructs a WFST of all possible translations of the source sentence span in each cell. When a rule contains a non-terminal, a pointer to another cell is created. If a cell is pruned during the search, these pointers are expanded. After all cell WFSTs are created, the top-most cell, which covers the entire source sentence span, is expanded into a translation WFST encoding all possible translations of the source sentence. Finally, the translation WFST is rescored by composing it with a language model WFST.

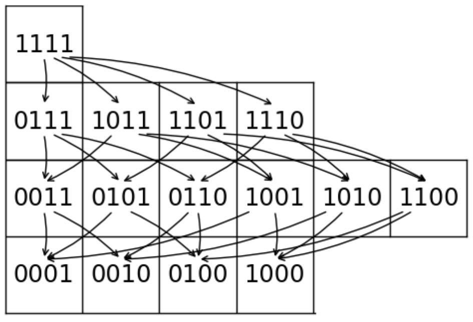
Constructing and operating on the WFSTs is accomplished using standard FST operations, including determinization, minimization, composition, and replacement. The best translation is found using the shortest-path operation. The use of standard FST operations, provided by an external library, makes the decoder implementation simple (Iglesias et al., 2009). Additionally, experiments in Iglesias et al. (2009) and de Gispert et al. (2010) show that HiFST requires less pruning and therefore results in fewer search errors and improved translation performance.

5.2 Graph coverage

In Hierarchical Phrase-Based Translation, a rule coverage denotes the part of the source sentence translated by the rule. A rule coverage is expressed in terms of a token span. For example, rule $R_{1:3} : X \leftarrow \langle s_1 X_0, t_1 X_0 \rangle$ translates tokens 1 to 3 of the source sentence. During Hiero decoding, rules are grouped by their coverage in a modified CYK grid, as



(a) Hiero modified CYK grid for a source sentence of length five.



(b) Graph coverage grid for a graph with four nodes. The arrows demonstrate the cell coverage hierarchy.

Figure 5.1: Examples coverage grids based on token spans (a) and graph coverage (b).

shown in Figure 5.1a. Each cell in the CYK grid refers to a token span over the source sentence, based on its horizontal (starting token index) and vertical position (length of the token span).

Non-terminal symbols inside rules are also assigned coverages. For example, in the above rule $R_{1:3}$, the non-terminal X_0 covers the source token sequence from the second to the third token. In the CYK grid, rules with non-terminal symbols point to cells lower in the grid, indicating the non-terminal's coverage and the rules that could be nested in the non-terminal's place. The arrow represents a rule's non-terminal symbol pointing to a cell covering a subspan of its coverage.

Token spans, however, cannot be used to express the coverage of a graph. Instead, in this section I define **graph coverage** to denote the part of the source graph translated by a graph-to-string rule. A graph coverage C_G of graph $G = (V, E)$ consists of the covered set of nodes $C_G \subseteq V$.

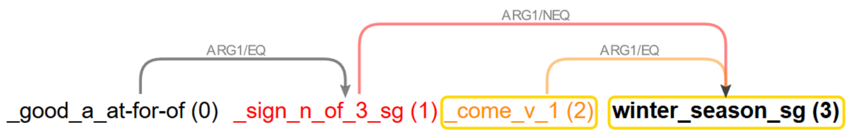
There exists a hierarchy of coverages of graph G. Namely, C_{G_1} is higher in the hierarchy than C_{G_2}, $C_{G_1} > C_{G_1}$, if C_{G_2} is a proper subset of C_{G_1}, $C_{G_2} \subset C_{G_1}$. Like subset, hierarchy is a transitive property.

Since it is cumbersome to refer to a coverage as a subset of the node set, I use **bit vector coverage** as a concise representation of graph coverage. Assuming a bijective function $f : V \to O$, where $O = \{o; 1 \leq o \leq |V|\}$, which creates an ordering over the nodes of graph G, graph coverage C_G can be represented as a bit vector of length $|V|$. Each node v of graph G has a corresponding position in the bit vector BV:

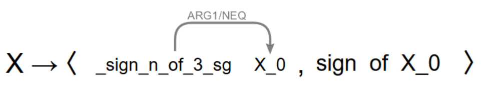
$$BV[f(v)] = \begin{cases} 1, & \text{if } v \in C_G \\ 0, & \text{if } v \notin C_G \end{cases} \tag{5.1}$$

A particular position in the bit vector has a value of 1 if the node is in the covered node set, otherwise its value is 0. An example of a hierarchy of coverages for a graph with four nodes is shown on a grid in Figure 5.1b.[2] The hierarchy of coverages can also be stated as a bitwise comparison of coverage bit vectors: If, for every v in V:

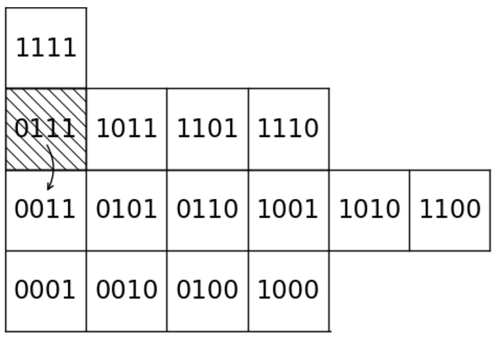
[2]Note that hierarchy relationship is transitive and therefore the exhaustive relationships are not shown in the figure.



(a) An example input graph coverage. The covered nodes are highlighted, while their node IDs are shown in parentheses.



(b) An example rule.



(c) Coverage grid for graph shown in (a). A cell's vertical position represents the number of nodes covered.

Figure 5.2: A graph coverage example. An example graph in (a) is covered by the rule in (b). Assuming that the node labels represent the ordering function, the rule coverage can be expressed as the bit vector $01X_0X_0$. The coverage grid for graph in (a) is shown in (c). The shaded cell contains the rule shown in (b). The arrow points to the coverage of rule's non-terminal symbol lower in the hierarchy.

$$BV_{G_1}[f(v)] \lor BV_{G_2}[f(v)] \Leftrightarrow BV_{G_1}[f(v)] \tag{5.2}$$

is true, $BV_{G_1} > BV_{G_2}$. A non-terminal symbol X in a rule R with graph coverage C_{G_R} has an associated graph coverage C_{G_X} such that C_{G_X} is lower in the hierarchy compared to C_{G_R}, $C_{G_R} > C_{G_X}$. This constraint ensures that nested rules always cover a subset of nodes covered by the parent rule. Bit vector representation of graph coverage can be extended to represent not only the rule's graph coverage but also the coverages of rules' non-terminal symbols:

$$BV[f(v)] = \begin{cases} X_n, & \text{if } v \in C_{G_{X_n}} \\ 1, & \text{if } v \in C_G \\ 0, & \text{if } v \notin C_G \end{cases} \tag{5.3}$$

Similar to Hiero, graph-to-string rules can also be grouped in a grid according to their coverage. An example of rule's graph coverage is shown in Figure 5.2.

While the size of the CYK grid of word spans grows in $\mathcal{O}(n^2)$ space with input size, the size of the grid of graph coverages grows in $\mathcal{O}(2^n)$. In practice, however, the space of possible graph coverages of a given input graph is severely constrained by the rules present in the grammar.

5.3 Rule application

In this section, I describe rule application, the first part of decoding a DMRS graph into the target string. Rule application identifies (1) which rules from the translation grammar can be used to translate the input graph, and (2) the rules' corresponding graph coverages (introduced in the previous section).

The synchronous context-free grammar constructed with the rule extraction algorithm (introduced in Section 4.2) consists of terminal and non-terminal graph-to-string rules. In the case of terminal rules, a simple rule application algorithm would take every terminal rule in the grammar and determine whether it can apply it to the input DMRS graph, by checking whether rule's source side is a subgraph that respects the semantic parse. However, determining whether a graph is a subgraph of another graph is a well-known NP-complete problem of subgraph isomorphism. In the case of non-terminal rules, the additional difficulty of rule application stems from non-terminal symbols, which can cover different parts of the graph.

To address both the problem of subgraph isomorphism and non-terminal rules, I propose a rule application algorithm that (1) emulates rule extraction to generate terminal and non-terminal rule source sides that could be used to decode the input graph, and (2) searches the grammar for rules with matching source sides.

The proposed rule application algorithm formulation avoids the subgraph isomorphism problem. Instead, it needs to address its special case, the *graph* isomorphism problem, where the latter is not known to be NP-complete (Read and Corneil, 1977). As discussed in Section 4.3.1, the graph isomorphism problem can be solved via graph canonization. Namely, determining whether two graphs are isomorphic is accomplished by finding a canonic representation for each of them and comparing the two representations: if they are the same, the graphs are isomorphic. In the case of rule application, finding a pair of isomorphic graphs means that an applicable rule for the given input graph is found in the grammar.

I presented a heuristic solution for graph canonization in Section 4.3.1 for the purpose of aggregating rules for grammar construction. I use the same solution for the rule application algorithm described below. In particular, the source-side graphs generated for the input graph are transformed into their canonical representations prior to querying the grammar, whose rules are already in a canonical representation. The above described formulation is well-suited to processing in a distributed computational environment and therefore viable for use with large grammars in practice (see discussion in Section 1.5).

In the remainder of this section, I present the rule application algorithm and demonstrate its operation on an example.

Alg. 5.1 **RuleApplication**$(G, \mathcal{R}, N, V_{max})$ is the algorithm which, given graph G and grammar $\mathcal{R}$, computes $\mathcal{R}_G$ - the rules from grammar $\mathcal{R}$ applicable to graph G, and for every applicable rule its possible coverages of graph G. Parameter N controls the maximum number of non-terminals allowed in a rule; parameter V_{max} controls the maximum node set size of the source-side subgraph. The parameter values should match the ones used in the rule extraction algorithm presented in Section 4.2.4.

1. (Create terminal subgraphs.) Create the set of terminal subgraphs of G, $\mathcal{G}_T \leftarrow$ **CreateSemanticSubgraphs**(G) (Algorithm 4.1 on page 67).

2. (Compute terminal subgraph coverages.) For each subgraph g in $\mathcal{G}_T$, compute its coverage of graph G.

3. (Initialize variables.) Initialize current working set $\mathcal{G}_{ws} \leftarrow \mathcal{G}_T$. Initialize result set $\mathcal{G}_{res} \leftarrow \mathcal{G}_T$.

4. (Repeat.) Repeat for N iterations to create source-side graphs with up to N non-terminals. Keep track of iteration number with $i \leftarrow 0$, incrementing it after each loop.

   (a) (Initialize new working set.) Set $\mathcal{G}_{new} \leftarrow \{\}$.

   (b) (Find candidate graph pairs.) For every graph $g \in \mathcal{G}_{ws}$, find terminal graphs $g_T \in \mathcal{G}_T$, such that g_T is a subgraph of graph g, $V(g_T) \subset V(g)$ and $E(g_T) \subset E(g)$. Determining the subgraph property is a trivial problem since both graphs are derived from the same initial graph G. Each such pair of graphs (g, g_T) is a candidate for creation of a non-terminal graph.

   (c) (Create candidate non-terminal graphs.) For every candidate pair of graphs (g, g_T), create a non-terminal graph g_{NT} by calling **CreateNonterminalGraph**(g, g_T, i) (Algorithm 4.3 on page 69). Additionally, for each g_{NT}, compute non-terminal coverage of graph G.

   (d) (Apply non-terminal graph constraints.) For every candidate non-terminal graph g_{NT}, check that it conforms to **non-terminal graph constraints** (defined in Section 4.2.3). Non-terminal graphs that conform to constraints are collected in $\mathcal{G}_{new}$.

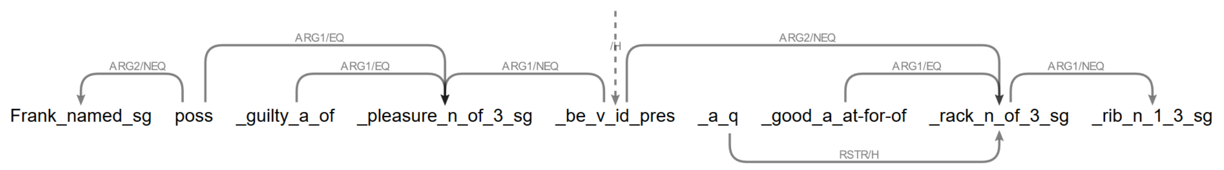
   (e) (Update variables.) Extend result set $\mathcal{G}_{res}$ with $\mathcal{G}_{new}$. Set $\mathcal{G}_{ws} \leftarrow \mathcal{G}_{new}$. In the next iteration, the working rule set $\mathcal{G}_{ws}$ will contain newly created non-terminal graphs with an additional non-terminal symbol.

5. (Filter and canonicalize.) Filter result graph set $\mathcal{G}_{res}$ based on V_{max} and transform remaining graphs into their canonical form (using heuristic solution described in Section 4.3.1).

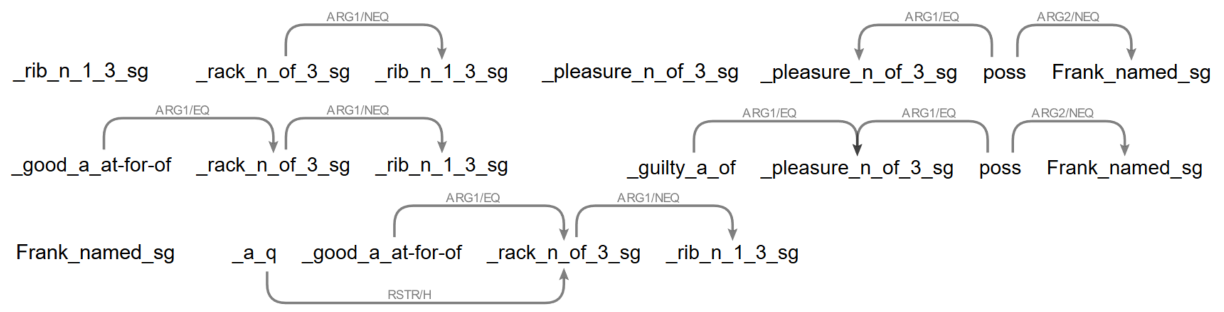
6. (Query grammar.) Intersect the set of graphs $\mathcal{G}_{res}$ with rules' source-side graphs of grammar $\mathcal{R}$ to obtain applicable rules $\mathcal{R}_G$ and associated coverages of G: $\mathcal{R}_G = \{(r, C_G(g)) \mid r = \langle g, e, \sim \rangle, r \in \mathcal{R} \text{ and } g \in \mathcal{G}_{res}\}$

7. (Return.) Return $\mathcal{R}_G$.

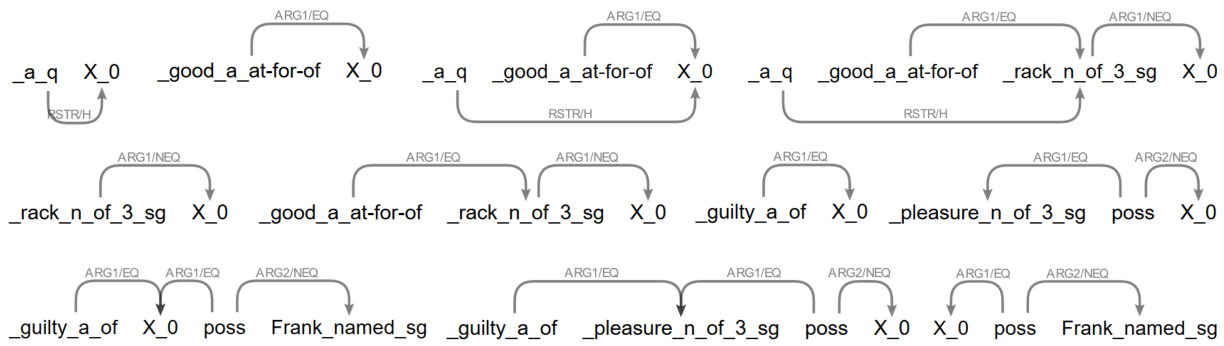
Let's look at an example of rule application output for the sentence "Frank's guilty pleasure is a good rack of ribs." The algorithm was set to produce source-side graphs with at most two non-terminal symbols and at most five nodes ($N = 2, V_{max} = 5$). Figure 5.3 shows the DMRS graph of the sentence at the top, and the source-side graphs created by the rule application algorithm (after filtering and canonization step) at the bottom. The source-side graphs are grouped according to the number of non-terminals they contain - zero, one, or two. This grouping reflects the order in which the rule application algorithm constructs the source-side graphs: terminal source-side graphs (no non-terminals, corresponding to step 1 of the algorithm) are constructed first, followed by non-terminal graphs with increasing number of non-terminals (step 4 of the algorithm).



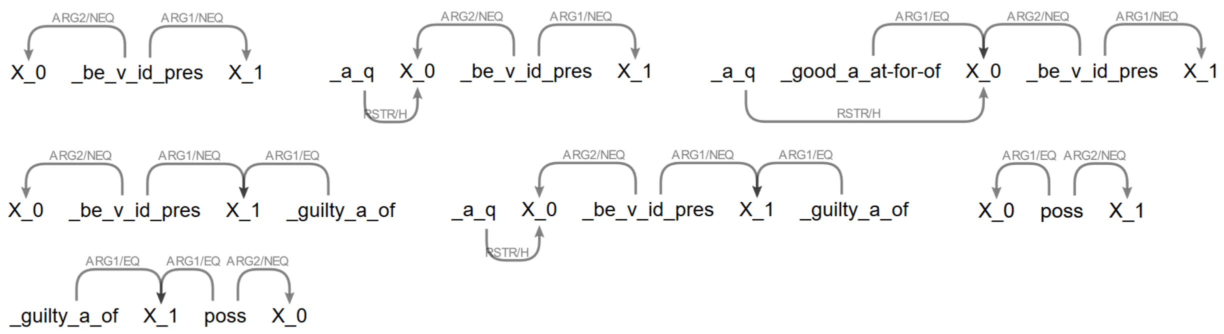
(a) Input DMRS graph.



(b) Terminal source-side graphs.

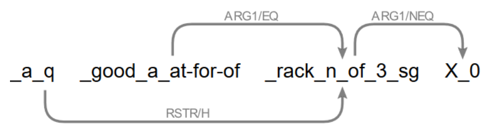


(c) Non-terminal source-side graphs (1 non-terminal).



(d) Non-terminal source-side graphs (2 non-terminals).

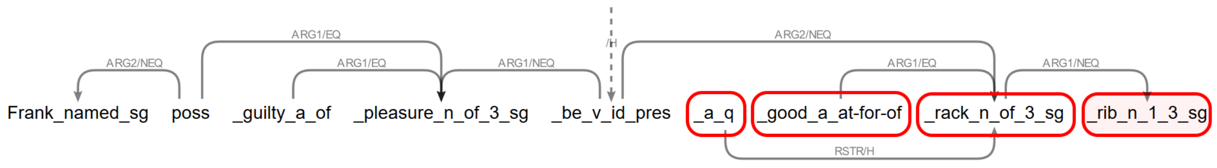
Figure 5.3: Rule application output for the example sentence "Frank's guilty pleasure is a good rack of ribs." At the top of the figure is the input DMRS graph representing the example sentence. The remaining three parts of the figure show the filtered output of the rule application algorithm at three iterations of the algorithm loop (statement (4)), increasing the number of non-terminal symbols in the extracted source-side graphs. Note that coverages, which are part of the algorithm output, are omitted from the figure for clarity.



(a) Source-side graph created by the rule application algorithm (shown in Figure 5.3).

$0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ X_0$

(b) Coverage output by the rule application algorithm for the graph in (a).



(c) Input DMRS graph covered by the source-side graph shown in (a) (non-terminal coverage is shaded).

Figure 5.4: Rule application coverage output example for a single source-side graph.

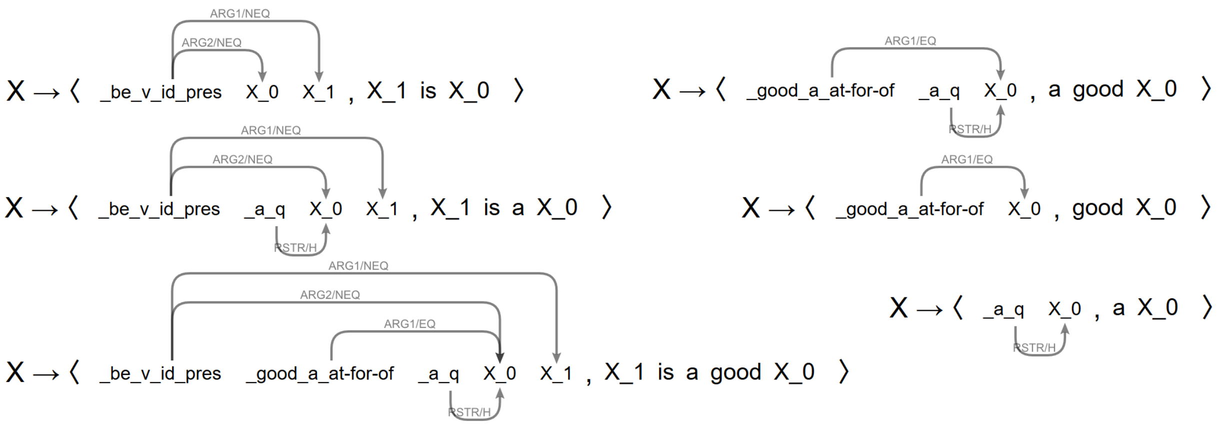

X → ⟨ _be_v_id_pres X_0 X_1 , X_1 is X_0 ⟩
ARG1/NEQ
ARG2/NEQ
X → ⟨ _good_a_at-for-of _a_q X_0 , a good X_0 ⟩
ARG1/EQ
RSTR/H
X → ⟨ _be_v_id_pres _a_q X_0 X_1 , X_1 is a X_0 ⟩
ARG1/NEQ
ARG2/NEQ
RSTR/H
X → ⟨ _good_a_at-for-of X_0 , good X_0 ⟩
ARG1/EQ
X → ⟨ _be_v_id_pres _good_a_at-for-of _a_q X_0 X_1 , X_1 is a good X_0 ⟩
ARG1/NEQ
ARG2/NEQ
ARG1/EQ
RSTR/H
X → ⟨ _a_q X_0 , a X_0 ⟩
RSTR/H

Figure 5.5: The set of six applicable rules returned by the rule application algorithm for the input DMRS graph shown in Figure 5.3a given the grammar consisting of rules shown in Figure 4.10.

Note, however, that only filtered source-side graphs are shown in Figure 5.3 (i.e., the set $\mathcal{G}_{res}$ created after step 5). There are additional source-side graphs that are created during the algorithm execution that contain more than five nodes, but are subsequently filtered. For example, one of the terminal graphs that is later filtered out is the input DMRS graph itself (shown at the top of Figure 5.3).

The output of rule application algorithm are both source-side graphs and their coverages of the input graph. For clarity, Figure 5.3 did not show the coverages. Instead, I show the coverage output for an example source-side graph in Figure 5.4.

In the final step of the rule application algorithm (before returning the output), the filtered source-side graphs are used to query the grammar $\mathcal{R}$ in order to obtain the set of applicable rules. Let's assume that the grammar $\mathcal{R}$ consists of rules shown in Figure 4.10, which were extracted from a single example shown in Figure 4.2.[3] Intersecting the example grammar $\mathcal{R}$ with the filtered source-side graphs shown in Figure 5.3 yields a set of six applicable rules. They are shown in Figure 5.5.

[3]Of course, in practice, the grammar will be constructed from rules extracted from millions of examples.

5.4 Decoding

In the previous section, I described rule application as a procedure that, for a given graph we wish to decode, identifies applicable rules and organizes them in a grid according to their coverage. In this section I describe the procedure for creating the hypothesis space and selecting the best hypothesis given the rule application output.

The hypothesis space is represented as a Weighted Finite State Acceptor (WFSA). A WFSA is an automaton in which each transition (arc) between two states has (1) an input label and (2) a weight that indicates the cost of using the arc. In a Weighted Finite State Transducer (WFST) an arc has an output label in addition to the input label. All automata discussed in this chapter are weighted, unless stated otherwise. We therefore adopt terms FSA and FST to mean weighted automata. Finally, Recursive Transition Network (RTN) is an automaton based either on FSA or FST that can contain non-terminal arc labels pointing to other automata.

When constructing the hypothesis space, every rule's target side and weight are represented as an RTN. RTNs representing rules with the same coverage are grouped into cell RTNs so that the cell RTN accepts every rule's target side. The non-terminal arc label referring to another RTN is an encoding of that RTN's graph coverage. In order to create the FSA representing the hypothesis space, a replacement algorithm starting at the top cell (the one covering the largest part of the input graph) recursively replaces RTN's non-terminal labels with corresponding automatons. Finally, the FSA is composed with the language model FSA to add the language model scores. The best hypothesis is then selected as the shortest path through the hypothesis space FSA.

The work described in this section is in large part a reimplementation and adaptation of the ideas behind HiFST decoder introduced in Section 5.1. Our implementation of the decoder uses the OpenFST library[4] (Allauzen et al., 2007) and pyFST library (Chahuneau, 2015), a Python interface to OpenFST, to construct and operate on automata. In the remainder of the chapter I adopt the mathematical notation used by de Gispert et al. (2010) and Allauzen et al. (2014) to formally describe automata and the hypothesis space construction and manipulation.

5.4.1 Finite state transducers

In this section I formally define semirings, weighted finite state transducers, and recursive transition networks, which are used in constructing the hypothesis space. The definitions are based on Allauzen et al. (2007, 2014).

A **semiring** $(\mathbb{K}, \oplus, \otimes, \bar{0}, \bar{1})$ is a set $\mathbb{K}$ with generalizations of binary addition $\oplus$, which is associative, commutative, and has the zero element $\bar{0}$ as identity, and multiplication $\otimes$, which is associative, has the unit element $\bar{1}$ as identity, has $\bar{0}$ as annihilator, and distributes with respect to $\oplus$ (Cohn, 2003; Allauzen et al., 2007, 2014). A semiring differs from a ring in that it may lack negation (Allauzen et al., 2014).

A **weighted finite state transducer (WFST)** is a tuple $T = (\Sigma, \Omega, \mathcal{Q}, \mathcal{I}, \mathcal{F}, E, \lambda, \rho)$ defined over semiring $\mathbb{K}$, where Σ is a finite input alphabet, Ω is a finite output alphabet, $\mathcal{Q}$ is a finite set of states, $\mathcal{I}$ is a set of initial states, $\mathcal{I} \subseteq \mathcal{Q}$, $\mathcal{F}$ is a set of final states,

[4] http://www.openfst.org

$\mathcal{F} \subseteq \mathcal{Q}$, E is a finite set of transitions between two states with input and output labels, $E \subseteq \mathcal{Q} \times (\Sigma \cup \{\varepsilon\}) \times (\Omega \cup \{\varepsilon\}) \times \mathbb{K} \times \mathcal{Q}$, λ is initial state weight assignment, $\lambda : \mathcal{I} \to \mathbb{K}$, and ρ is final state weight assignment, $\rho : \mathcal{F} \to \mathbb{K}$ (Allauzen et al., 2007). ε is the empty string which as an input label denotes a transition which can be taken without consuming a symbol or as an output label does not output a symbol (Mohri, 2002).

A special case of WFST without output labels is **weighted finite state acceptor (WFSA)**. WFSA is a tuple $A = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{F}, \mathcal{E}, \lambda, \rho)$, where $E \subseteq \mathcal{Q} \times (\Sigma \cup \{\varepsilon\}) \times \mathbb{K} \times \mathcal{Q}$.

A **recursive transition network (RTN)** is a tuple $R = (\Sigma, \mathcal{N}, T_n, \mathcal{S})$ defined over semiring $\mathbb{K}$, where Σ is a finite input alphabet, $\mathcal{N}$ is a finite alphabet of non-terminal symbols, T_n is a family of WFSTs with input alphabet $\Sigma \cup \mathcal{N}$ and $n \in \mathcal{N}$, and $\mathcal{S}$ is the root non-terminal, $\mathcal{S} \in \mathcal{N}$.

In the remainder of this chapter, I use $\mathcal{L}_T$, $\mathcal{L}_A$, and $\mathcal{L}_R$ to refer to WFSTs, WFSAs, and RTNs respectively.

5.4.2 Hypothesis space construction

The input to construction of hypothesis space is a coverage grid $\mathbf{R}$, consisting of cells $\mathbf{R}(C_G)$ which group rules according to their coverage. For each rule $r = \langle g, e, \sim \rangle, r \in \mathbf{R}(C_G)$, an RTN $\mathcal{L}_R(C_G, r)$ is constructed to represent the rule's target-side sequence S. $\mathcal{L}_R(C_G, r)$ is a single-path acceptor constructed by concatenating acceptors $\mathcal{L}_R(C_G, r, i)$ corresponding to elements of $S = S_1 ... S_{|S|}$. If S_i is a terminal symbol, it is used as arcs' input label. If S_i is a non-terminal symbol, its input label is a pointer to a graph coverage lower in the hierarchy. Therefore:

$$\mathcal{L}_R(C_G, r) = \bigotimes_{i=1...|S|} \mathcal{L}_R(C_G, r, i) \tag{5.4}$$

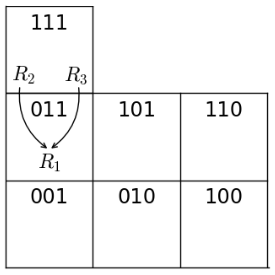
$$\mathcal{L}_R(C_G, r, i) = \begin{cases} \mathcal{A}(S_i) & \text{if } S_i \in \Sigma \\ \mathcal{L}_R(C'_G, r') & \text{otherwise} \end{cases} \tag{5.5}$$

where $\bigotimes$ is the concatenation operation and $\mathcal{A}(t), t \in \Sigma$ returns a single-arc automaton accepting symbol t (de Gispert et al., 2010). An example of terminal and non-terminal rules and their RTNs is shown in Figures 5.6b and 5.6c.

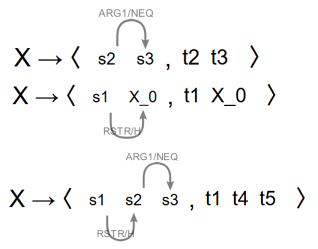
For each cell $\mathbf{R}(C_G)$, its RTN $\mathcal{L}_R(C_G)$ is constructed as a union of $\mathcal{L}_R(C_G, r), r \in \mathbf{R}(C_G)$:

$$\mathcal{L}_R(C_G) = \bigoplus_{r \in \mathbf{R}(C_G)} \mathcal{L}_R(C_G, r) \tag{5.6}$$

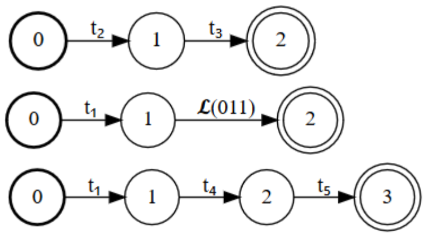
where $\bigoplus$ is the union operation. An example of a cell RTN is shown in Figure 5.6d. Finally, the replacement algorithm converts a recursive transition network into a finite state acceptor (Allauzen et al., 2014). Starting from the top cell $\mathcal{L}_R(C_{G_{top}})$ (cell which covers the largest part of the input graph, usually the cell that covers the entire graph), the algorithm recursively expands RTNs by replacing their non-terminal labels with corresponding automatons. The constructed FSA $\mathcal{L}_A$ encodes all hypotheses allowed by the grammar for the graph G. An example is shown in Figure 5.6e.



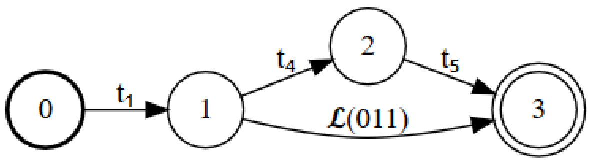
(a) Graph coverage grid containing three rules.



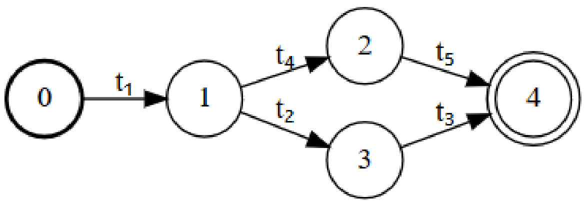
(b) From top to bottom, rules R_1, R_2, and R_3.



(c) From top to bottom, RTNs $\mathcal{L}_R(011, R_1)$, $\mathcal{L}_R(111, R_2)$, and $\mathcal{L}_R(111, R_3)$.



(d) Cell RTN $\mathcal{L}_R(111)$ created for coverage 111 from the union of $\mathcal{L}_R(111, R_2)$ and $\mathcal{L}_R(111, R_3)$ rule RTNs.



(e) The final FSA $\mathcal{L}_A$ created via the recursive replacement operation started from the top cell RTN $\mathcal{L}_R(111)$.

Figure 5.6: An example of construction of the hypothesis space FSA from rule application output (shown as a graph coverage grid in (a) and corresponding rules in (b)). Rule RTNs are constructed from rule target sides (c) and grouped according to their coverage to create cell RTNs (d). Finally, the final FSA is created via the replacement operation (e).

5.4.3 Selecting best hypothesis

The constructed FSA $\mathcal{L}_A$ can represent a large number of hypotheses. A mechanism for ranking them is needed to choose the best one. In FSA construction, a hypothesis is formed by applying a sequence of rules - a **derivation**. I use a log-linear model to score the derivations that generate the hypotheses in order to choose the best one. The log-linear model is the standard approach to scoring derivations in Hierarchical Phrase-Based Translation (Chiang, 2005, 2007; de Gispert et al., 2010). Following de Gispert et al. (2010), I use the tropical semiring defined as $(\mathbb{R} \cup \{\infty\}, \min, +, \infty, 0)$ when constructing the RTNs and for ranking the hypotheses in the final FSA. Due to the use of tropical semiring, rule probabilities p_r are transformed to rule costs, $c_r = -\log p_r$.

As described in Section 4.3.2, a number of features are computed for each rule. The feature importance is determined by its weight. Therefore, the cost of a rule c_r in a log-linear model is:

$$c_r = \sum_{f \in F} f(r)\lambda_f \tag{5.7}$$

where F is the set of feature functions, $f(r)$ is the value of feature f for rule r, and λ_f is the weight of feature f. Tuning of feature weights is discussed in Section 5.6.

Unlike rule features described so far, a language model score is not associated with a rule. Instead, an N-gram target language model scores target-side symbols based on $N-1$ previous symbols. Therefore, a language model score cannot be computed for a rule's target symbol sequence in isolation nor can it be computed over non-terminal symbols. For this reason, the language model scores are applied to the hypothesis FSA $\mathcal{L}_A$ via intersection (Allauzen et al., 2007) with the target language model FSA.[5] As with rule probabilities, language model probabilities are transformed to costs ($-\log$) for use with the tropical semiring.

The cost of a derivation d is the sum of the cost of its sequence of rules $r_1...r_N$, alongside the language model score:

$$\begin{aligned} c_d &= c_{lm}(e)\lambda_{lm} + \sum_{n=1}^{N} c_{r_n} \\ &= c_{lm}(e)\lambda_{lm} + \sum_{f\in F} \lambda_f \sum_{n=1}^{N} f(r_n) \end{aligned} \tag{5.8}$$

where e is the derivation hypothesis string, λ_{lm} is the language model weight and

$$c_{lm}(e) = -\log(p_{lm}(e)) \tag{5.9}$$

where p_{lm} is the language model. Selecting the best hypothesis from the FSA is therefore equal to finding the lowest cost derivation $\tilde{d}$:

$$\begin{aligned} \tilde{d} &= \arg\min_{d\in D} c_d \\ &= \arg\min_{d\in D} -\log(p_{lm}(e))\lambda_{lm} + \sum_{f\in F} \lambda_f \sum_{n=1}^{N} f(r_n) \end{aligned} \tag{5.10}$$

When constructing the rule RTN $\mathcal{L}_R(C_G, r)$, the rule cost c_r is associated with the final RTN state. According to the tropical semiring, the cost of a sequence of rule RTNs (i.e., derivation) is the sum of their individual rule costs. This corresponds to the definition of derivation cost in Equation 5.8. Therefore, the search for the best hypothesis in $\mathcal{L}_A$ corresponds to finding the shortest path through the FSA (intersected with the language model FSA) under the tropical semiring. The shortest path is found using the shortest path operation in OpenFST (Allauzen et al., 2007).

5.5 Improving decoding efficiency

The decoding procedure described in Section 5.4 can require significant time and computational resources for large input graphs and grammars. In this section I introduce two approaches to improve the decoding efficiency: in Section 5.5.1 I describe cell selection, a procedure which reduces the set of cell RTNs that need to be constructed without introducing any errors, while in Section 5.5.2 I describe local pruning, an approach that reduces the size of cell RTNs via pruning and allows a trade-off between quality and efficiency. The latter was previously used for the same purpose by the HiFST decoder.

[5] Language model application is done efficiently using the applylm tool of the HiFST toolkit (Blackwood et al., 2016)

5.5.1 Cell selection

In Section 5.4.2 I described the process of building cell RTNs based on rules and coverages resulting from rule application. However, not every cell ends up being used in the final FSA. Cell selection identifies such cells and avoids creating them in order to improve decoding efficiency. Unlike local pruning described in the following section, cell selection does not introduce any errors into the decoding process.

The result of rule application for a given graph are rules and their coverage of the input graph. These rules are grouped according to their coverage into cells, which are then encoded as RTNs. Subsequently, a replacement algorithm creates the FSA by recursively expanding cell RTNs, starting from the top cell (i.e., the one covering the largest part of the graph, usually covering the entire graph). In order for an RTN to be expanded and included in the FSA, it needs to be referenced by a non-terminal arc in one of the higher coverage RTNs, which themselves need to be referenced by the top cell. Therefore, any cell that is not referenced by the top RTN or one of its descendants, will not be expanded by the replacement algorithm. If we can identify such cells, we can avoid creating them. This is achieved by the **CellSelection** algorithm described below.

Alg. 5.2 **CellSelection**(C_T) is a recursive algorithm which, given a cell C_T, identifies the set of cells referenced by C_T or one of its descendants.

1. (Initialize selected cell set.) $\mathcal{C} \leftarrow \{C_T\}$.
2. (Recurse on directly referenced cells.) For every rule $r \in C_T$, identify non-terminal references c_i to lower coverage cells, and for every c_i update $\mathcal{C}$ with **CellSelection**(C_{c_i}).
3. (Return selected cell set.) Return $\mathcal{C}$.

Calling the **CellSelection** algorithm with the top cell gives us the set of cells that need to be created during hypothesis space construction (Section 5.4.2), avoiding creating cells that are not going to be used.

5.5.2 Local pruning

Using a non-trivial size grammar results in many competing rules that can be used to decode a part of the input graph. The RTNs representing these coverage cells can grow very large, containing millions of states and edges. When presented with such large RTNs, the recursive replacement algorithm fails to create the final FSA in reasonable time and space. This behaviour provides motivation for controlling the size of the search space using local pruning. Local pruning reduces the size of RTNs before they are expanded by the replacement algorithm by pruning away low scoring hypotheses. In this section, I describe the local pruning algorithm. Using local pruning can result in search errors as the optimal hypothesis may be pruned away before the search for shortest path is conducted on the final FSA. Because of this, I also describe a simple method for controlling the use of local pruning based on node coverage and number of states.

Before a cell RTN can be pruned, it needs to be expanded into an FSA in order to replace non-terminal transitions with terminal ones. However, the expanded FSA is still not complete, as it is missing the language model scores and pruning without language model scores could result in removing potentially high-scoring hypotheses. Because of this, language model scores need to be added to the cell FSA before pruning. However, they need to be removed after pruning so that the final FSA can still be intersected with the language model. Since not all cells are pruned (method for deciding which cells to prune is described later in this section), were language model scores not removed, some of them would include language model scores and some of them would not. The **LocalPrune** algorithm stated below addresses all of these issues.

LocalPrune$(c, \mathcal{M}, \mathcal{LM}, t, n)$ is the algorithm which, given a coverage c, a mapping $\mathcal{M}$ of coverages to their RTNs, and a language model $\mathcal{LM}$, prunes the RTN $\mathcal{M}(c)$ according to pruning threshold t and n-shortest paths. **Alg. 5.3**

1. (Create cell FSA.) $\mathcal{M}(c) \leftarrow \text{fstReplace}(\mathcal{M}(c), \mathcal{M})$. Expand cell RTN $\mathcal{M}(c)$ into an FSA with recursive replacement algorithm.

2. (Create cell LM FSAs.) Create two cell c language model FSAs, $\mathcal{LM}_{c+}$ and its counterpart with negative weights $\mathcal{LM}_{c-}$, by intersecting unweighted FSA $\mathcal{M}(c)$ with language model FSA $\mathcal{LM}$.

3. (Intersect cell and LM FSAs.) $\mathcal{M}(c) \leftarrow \text{fstIntersect}(\mathcal{M}(c), \mathcal{LM}_{c+})$. Intersect cell FSA with LM FSA to create a cell FSA with correct weights.

4. (Prune cell FSA.) $\mathcal{M}(c) \leftarrow \text{prune}(\mathcal{M}(c), t, n)$. Prune cell FSA by unioning two FSAs: (1) cell FSA pruned with threshold t and (2) cell FSA of n shortest paths.

5. (Remove LM weights.) $\mathcal{M}(c) \leftarrow \text{fstIntersect}(\mathcal{M}(c), \mathcal{LM}_{c-})$. Intersect cell FSA with LM FSA with negative weights to remove LM weights from the cell FSA.

6. (Return.) Return $\mathcal{M}(c)$.

The actual pruning of the cell FSA (step 4 of **LocalPrune** algorithm) consists of two parts. In part (1) OpenFST prune operation is used to remove all hypotheses that have a score that is more than threshold t worse than the score of the best hypothesis. However, the resulting FSA can vary in size significantly depending on the scale of scores. In order to avoid pruning the FSA too much, part (2) adds a constant number n of top hypotheses to the pruned FSA. This guarantees that the pruned cell will contain at least n hypotheses or remain unpruned if there are fewer than n hypotheses to begin with.

As indicated above, local pruning can introduce search errors. Therefore, it is desirable to only use local pruning when necessary, as opposed to applying it to every RTN in construction. In order to control when local pruning is applied to an RTN, its number of states and graph coverage are compared to a set of configured conditions. Namely, the conditions specify the minimum number of states for a given number of covered graph nodes which trigger local pruning. This control mechanism mirrors the mechanism used by HiFST decoder, which uses the number of states and the length of the word span instead. The pruning conditions are set experimentally based on decoding time and memory consumption.

5.6 Tuning

The graph-to-string decoder presented in Section 5.4 uses a log-linear model to score the hypotheses in order to choose the best one. Achieving the best possible translation quality requires the log-linear model parameters to be tuned. The standard tuning procedure of statistical machine translation systems is the Minimum Error Rate Training (MERT). MERT optimizes the model parameters by evaluating the translation performance on a tuning set (a set of sentences for which reference translations exist) in terms of the BLEU score (BLEU metric is described in Section 6.2.2).

MERT procedure optimizes the log-linear feature weights by varying them and observing the number of translation errors that occur, choosing the set of weights that produce the minimum number of errors. To achieve this, MERT requires N-best list of translation hypotheses to be produced for each sentence, as well as individual feature contributions of each rule used to create the hypothesis. However, the hypothesis space construction described above does not keep track of individual feature contributions. To remedy this, I adopt the tuning procedure described by de Gispert et al. (2010).

In Section 5.6.1 I describe decoding with alignment, which allows for keeping track of feature contributions of individual rules needed for MERT. Due to the increased computational needs of decoding with alignment, it is conducted with a constrained target space, previously generated by regular hypothesis space construction. Finally, I describe minimum error rate training and lattice minimum error rate training in the background Section 5.6.2.

5.6.1 Decoding with alignment

In hypothesis space construction (see Section 5.4.2) rule's target sides are encoded as input labels of FSA arcs. This enables FSAs to compactly represent the hypothesis space, since individual arcs can be shared by many rules and consequently many hypotheses. However, in doing so, it does not keep track of the rules that are used to construct a particular hypothesis.

Such FSA construction is sufficient for translating a sentence, as long as the arc scores are correct, which is ensured by using the tropical semiring. However, tuning the log-linear model parameters with MERT requires individual feature contributions to be computed for every hypothesis. As features are associated with rules, this means that for every hypothesis we need to know which set of rules was used to construct it (note that this is not as strong of a requirement as keeping track of hypothesis derivations).

To accommodate this requirement by MERT, I adapt decoding with alignment as described by de Gispert et al. (2010) in the HiFST decoder. In hypothesis space construction of decoding with alignment, Finite State Acceptors are replaced by Finite State Transducers. Output arc labels are used for target tokens, whereas the input labels are used to keep track of rule indices. The constructed rule FSTs can be viewed as a mapping from a rule id to a sequence of target tokens. For decoding with alignment, Equations 5.4 and 5.5 are redefined as:

$$\mathcal{L}_R(C_G, r) = \mathcal{A}_T(r, \varepsilon) \bigotimes_{i=1\ldots|S|} \mathcal{L}_R(C_G, r, i) \tag{5.11}$$

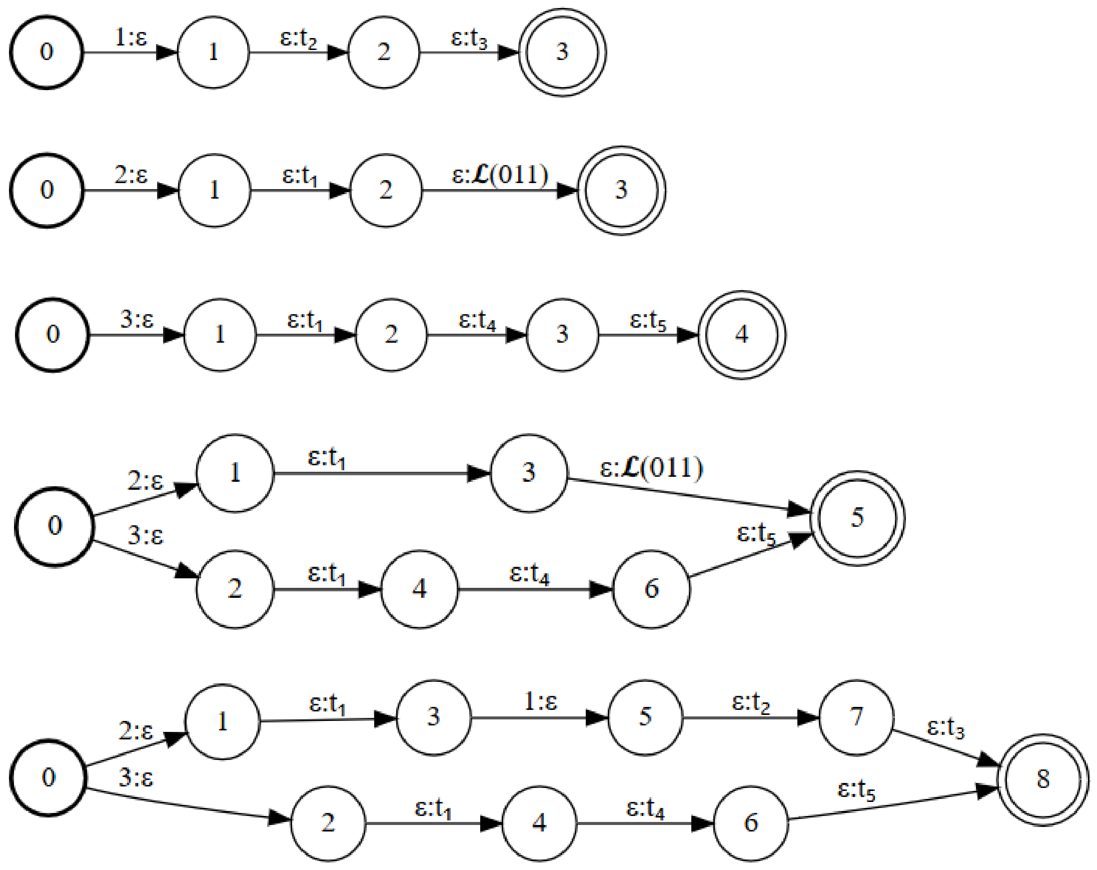

0 1:ε 1 ε:t2 2 ε:t3 3
0 2:ε 1 ε:t1 2 ε:𝓛(011) 3
0 3:ε 1 ε:t1 2 ε:t4 3 ε:t5 4
0 2:ε 1 ε:t1 3 ε:𝓛(011) 5; 0 3:ε 2 ε:t1 4 ε:t4 6 ε:t5 5
0 2:ε 1 ε:t1 3 1:ε 5 ε:t2 7 ε:t3 8; 0 3:ε 2 ε:t1 4 ε:t4 6 ε:t5 8

Figure 5.7: Decoding with alignment of the example from Figure 5.6. From top to bottom, rule RTNs $\mathcal{L}_R(011, R_1)$, $\mathcal{L}_R(111, R_2)$, $\mathcal{L}_R(111, R_3)$, cell RTN $\mathcal{L}_R(111)$, and final FST $\mathcal{L}_T$. The final FST encodes two hypothesis: (1) input rule id sequence `2,1` and output target token sequence `t1,t2,t3`; (2) input rule id sequence `3` and output target token sequence `t1,t4,t5`.

$$\mathcal{L}_R(C_G, r, i) = \begin{cases} \mathcal{A}_T(\varepsilon, S_i) & \text{if } S_i \in \mathbf{T} \\ \mathcal{L}_R(C'_G, r') & \text{otherwise} \end{cases} \tag{5.12}$$

where $\mathcal{A}_T(r, t), t \in \Sigma$ returns a single-arc transducer accepting the index of rule r in the input and the symbol t on the output (de Gispert et al., 2010).

A hypothesis in the final FST produced by decoding with alignment contains a sequence of target tokens on the output labels and a sequence of rules used to produce the target tokens on the input labels. An example of the hypothesis space constructed by decoding with alignment is shown in Figure 5.7. Note that the example is constructed from the same coverage grid shown in Figure 5.6b and that the final hypothesis space FST encodes the same two hypotheses on output labels as the final hypothesis space FSA shown in Figure 5.6e.

As stated above, FSAs are able to compactly represent the hypothesis space due to arc sharing by many hypotheses. In decoding with alignment, the potential for arc sharing and consequently compact representation of hypothesis space is significantly reduced: two hypotheses can share an arc only if they used the same rule to produce it. This can be seen in the final FST of example in Figure 5.7, which has 8 nodes and 9 arcs compared to 5 nodes and 5 arcs of FSA in Figure 5.6e, despite encoding the same target token hypotheses. Consequently, the hypothesis space produced by decoding with alignment can grow significantly larger, making it computationally infeasible for many sentences. To address this problem, de Gispert et al. (2010) propose a three-step decoding procedure for tuning with MERT, which I adopt here:

1. Construct the hypothesis space FSA (as described in Section 5.4.2) and prune it so that N-best hypotheses are left in the FSA.
2. Construct the hypothesis space FST (as described in current section) with a target space constrained to N-best FSA from the previous step.
3. Convert the hypothesis space FST to an FSA with feature contributions as arc labels.

Constraining the target space in step 2 is achieved using a full and substring reference acceptors during decoding with alignment. The full reference FSA accepts only the N-best hypotheses from step 1 of the decoding procedure. The substring reference FSA in addition also accepts any substring of the same hypotheses. The details on construction of reference FSAs based on N-best hypothesis FSA are described in de Gispert et al. (2010).

The target space of decoding with alignment is constrained by composing the final FST with the full reference FSA. This means that only top N hypothesis derivations are kept in the FST. However, the efficiency problem remains unaddressed for RTNs used to construct the hypothesis space. To remedy this problem, RTNs are constrained by composing them with the substring reference FSA, which bears similarity to local pruning described in Section 5.5.2.

Finally, in step 3 of the decoding procedure, individual feature contributions towards each hypothesis are identified. This is achieved by composing the FST constructed in step 2 with a mapping FST which maps sequences of rule indices to their feature values. The feature values are represented as a vector of floating point numbers. The mapping FST is constructed from the grammar and takes advantage of the tropical sparse tuple weight semiring, which allows it to store vectors of feature values as arc labels.[6]

5.6.2 Lattice Minimum Error Rate Training

Minimum Error Rate Training (MERT), introduced by Och (2003), is a procedure for tuning parameters of a log-linear SMT model using a non-differentiable objective function. MERT superseded other tuning procedures using maximum likelihood and related criteria because it enables the use of automatic evaluation metrics, such as BLEU, as training criteria. These alternative training criteria enable direct optimization of translation quality and better reflect the performance on unseen test data.

Given source sentences f_1^S and reference translations r_1^S, the SMT system produces N-best lists of candidate translations $K_s = \{e_{s,1}, ..., e_{s,N}\}$. The number of errors for a translation e with respect to the reference translation is measured with function $E(r, e)$. Assuming errors for individual sentences can be summed to produce corpus error count,

[6]I use the HiFST alilats2splats tool to accomplish the third step of the decoding procedure.

$E(r_1^S, e_1^S) = \sum_{s \in S} E(r_s, e_s)$. The goal of MERT is then to find a set of parameters λ_F such that the number of translation errors across the corpus is minimized:

$$\hat{\lambda}_F = \underset{\lambda_F}{\arg\min} \sum_{s \in S} E(r_s, \hat{e}(f_s; \lambda_F)) \tag{5.13}$$

where the translation candidate from N-best list with lowest derivation cost under current parameters is chosen:

$$\hat{e}(f_s; \lambda_F) = \underset{e \in K_s}{\arg\min}\, c_d(e; \lambda_F) \tag{5.14}$$

The basic variant of MERT chooses a single parameter $\lambda_{f \in F}$ to optimize at a time by varying its values alongside a line in the parameter space. For each sentence candidate set K_s, an upper envelope of scores at different values of the parameter $\lambda_{f \in F}$ is computed. The upper envelope encodes all possible outcomes of varying a single parameter for the given sentence and can be projected onto error counts of corresponding translation candidates. Corpus wide error counts are obtained by merging sentence upper envelopes and the corresponding error counts. Optimal value of parameter $\lambda_{f \in F}$ can then be found by traversing the corpus error surface and finding the minimum. After parameters are updated, new sentence candidate translations are computed and the process is repeated (Och, 2003; Macherey et al., 2008).

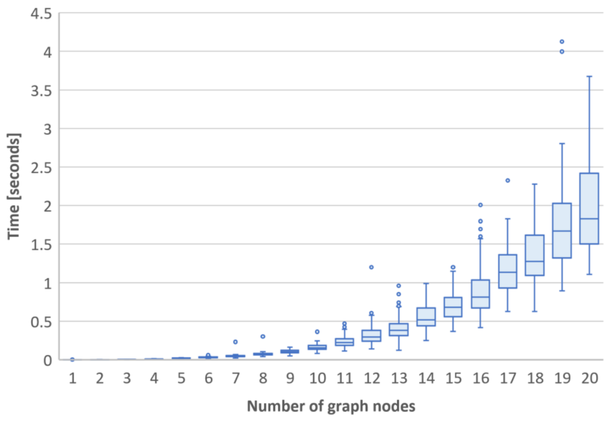
Lattice Minimum Error Rate Training (LMERT) is an extension of MERT that yields faster tuning convergence and explores a larger space of candidate translations (Macherey et al., 2008). Whereas MERT uses an N-best list of translation candidates for each sentence, LMERT uses a lattice. As N-best lists capture only a small part of the search space, the number of tuning iterations that require retranslating sentences after parameter update is large. On the other hand, lattices encode a significantly larger number of translation candidates, which in turn reduces the number of iterations until parameter convergence. The two candidate representations require different algorithms to compute the upper envelope: MERT uses the SweepLine algorithm, while LMERT uses the Lattice Envelope algorithm (both algorithms are described in Macherey et al. (2008)).

I used the LMERT implementation included with the HiFST SMT system[7] for tuning system parameters presented in this thesis. The HiFST LMERT implementation is based on the work by Waite et al. (2011), who model the optimization as computing the shortest distance in a Weighted Finite-State Transducer with Tropical Polynomial Weights.

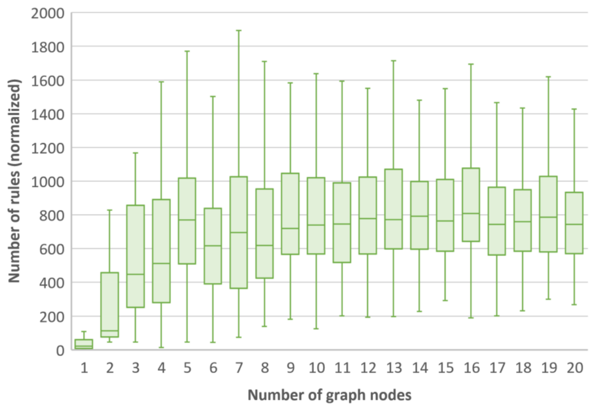
5.7 Decoding performance analysis

In previous sections I described rule application and decoding components. In this section I demonstrate their performance on a representative dataset in terms of running time (rule application and decoder) and number of rules produced (rule application). These experiments demonstrate the viability of the system presented in this thesis for large-scale translation and realization tasks. It is important to note, however, that the system implementation (partly described in Section 1.5) is not optimized or mature. The results

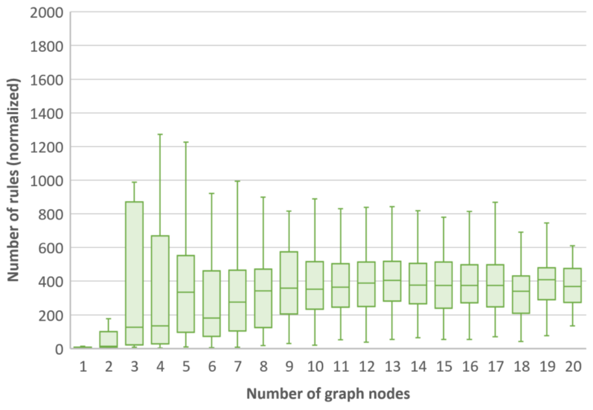
[7]https://ucam-smt.github.io/



(a) Rule application sentence running time distribution (RuleApplication algorithm steps 1-5).



(b) Translation task rule application rules.



(c) Realization task rule application rules.

Figure 5.8: Rule application runtime (top row) and number of applicable rules (bottom row) for 1774 sentences grouped by number of nodes. The number of rules is normalized by the number of graph nodes. I do not show outliers in (b) and (c) in order to preserve clarity.

should therefore only give an indication of what performance can be achieved, but not its limits. I begin the analysis with rule application in Section 5.7.1 and finish with decoding in Section 5.7.2.

5.7.1 Rule application

I evaluate the rule application performance in terms of running time and number of rules produced per sentence. The applied rules are subsequently used by the decoder, whose performance I evaluate in the following section (Section 5.7.2).

Rule application performance is evaluated on newstest2013 dataset filtered to DMRS graphs with 20 or fewer nodes, yielding 1774 sentences (this dataset is used for tuning in Chapters 6 and 7). Both translation and realization grammars have been estimated on the same corpora (WMT15 en-de translation corpora, described in Section 6.2.1). The translation grammar contains 15.1 million rules, whereas the realization grammar contains 26.9 million rules. The rule application algorithm uses the same parameter values as were used for estimating the grammars. Both translation and realization rule application therefore use $N = 2$ and $V_{max} = 5$.

I evaluate rule application algorithm running time per sentence for algorithm steps 1-5 (RuleApplication algorithm in Section 5.3). The performance of RuleApplication algorithm steps 1-5 is identical for translation and realization if performed on the same dataset. Consequently, I report running times only once, in Figure 5.8a. The algorithm for each sentence was run on a single CPU core.

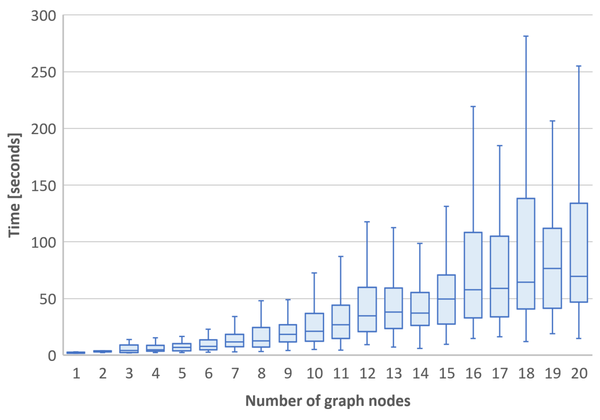
Figure 5.8a is a box and whisker chart showing distribution of sentence running times at each graph node set size. Outliers (sentence running times more than 1.5 times the interquartile range, $IQR = Q_3 - Q_1$ or the length of the box, away from either end of the box) are shown as individual data points. I will use this type of chart to present decoding performance in the remainder of the section.

Based on Figure 5.8a we can conclude that the running time of steps 1-5 of RuleApplication algorithm increase faster than linearly. The degree of dispersion (spread) of sentence running time increases with the number of graph nodes representing a sentence. The longest sentence running time is 4.12 seconds for a sentence with 19 nodes. Compared to decoder sentence running times reported in the following section, rule application presents a minor part of the total time needed to decode a sentence.

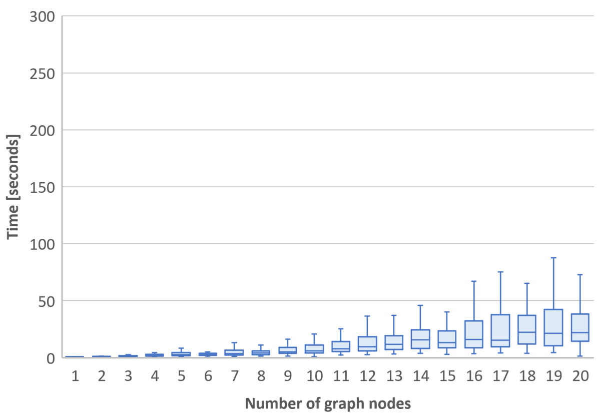
The final algorithm step (step 6), querying the grammar, is performed for the entire set of sentences with Apache Spark operations (Section 1.5). Therefore, its individual sentence running times cannot be reported. Instead, I report the total running time of rule application for the entire dataset. The complete rule application (steps 1-6) was run on 1774 sentences with Apache Spark on a single machine with the following settings: 16 CPU cores, 10 GB driver RAM limit, 12 GB executor RAM limit, and 50 partitions. In addition to steps 1-6 of rule application algorithm, glue, CARG, and disconnected rules (introduced in Section 6.1) were created for each sentence for both translation and realization tasks. Mapping rules were also created for translation task. The total running time was 1077 seconds (17 minutes 57 seconds) for translation task and 994 seconds (16 minutes 34 seconds) for realization task.

Figures 5.8b and 5.8c show the number of resulting applicable rules for translation and realization tasks respectively. The number of applied rules is normalized by the number of sentence graph nodes. We can observe that, apart from graphs with five nodes or fewer, the number of applicable rules per node found by rule application algorithm is constant. I theorize that the variability of applicable rules per node for graphs with five nodes or fewer occurs due to two related reasons: (1) graphs with five nodes or fewer often represent commonly used phrases and may have therefore been observed in the training corpora, yielding additional rules translating the input graph as a whole; (2) the RuleApplication algorithm parameter $V_{max} = 5$ specifies that the maximum number of nodes in a rule's source side is five, which means that rules that fully translate these graphs can exist, in addition to more granular rules translating parts of them.

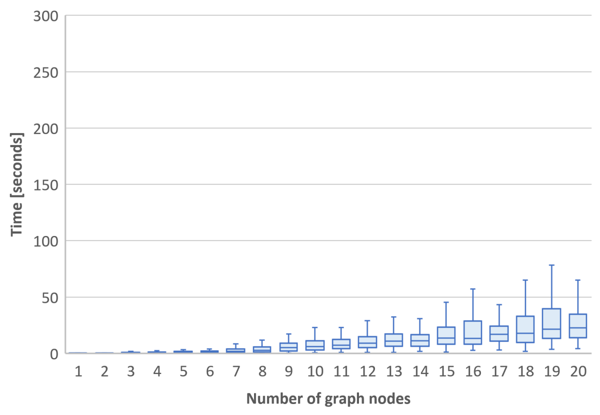
Figures 5.8b and 5.8c additionally show there is a marked difference in the number of applicable rules per node between translation and realization tasks. Whereas realization grammar yields around 400 applicable rules per graph node, translation grammar yields around double that amount. I hypothesize that the difference is the result of different grammar composition arising from the noise introduce by SMT alignments (described in Section 3.3.3). In particular, I observed that translation task applicable rules are dominated by rules translating very common graph fragments, such as determiners (for example, corresponding to `the X_0`).



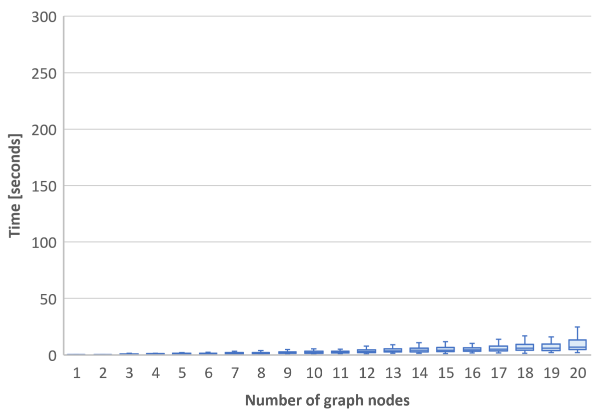
(a) Translation task in translation mode.



(b) Translation task in alignment mode.



(c) Realization task in translation mode.



(d) Realization task in alignment mode.

Figure 5.9: Decoding runtime for 1774 graphs (filtered newstest2013 of graphs with up to 20 nodes, described in Chapter 6) grouped by number of nodes.

5.7.2 Decoder

I evaluate decoder performance in terms of running time, comparing translation mode decoding (described in Section 5.4) and alignment mode decoding (described in Section 5.6.1) for translation and realization tasks. Translation mode decoding includes hypothesis space construction, language model application, and n-best shortest-path search. On the other hand, alignment mode decoding includes only hypothesis space construction, constrained to the n-best results of translation mode.

Decoding performance is evaluated on the same newstest2013 dataset as rule application. Additionally, it uses the applied rules produced for translation and realization tasks in the previous section. I use a German 4-gram language model estimated on monolingual corpora and target side of parallel corpora (details are described in Section 6.2.1) for translation mode of translation task. I use an English 4-gram language model for the equivalent realization task (details in Section 7.1.1).

Translation mode hypothesis space construction of both translation and realization tasks is pruned with the following setting: (8, 20, 9), (5, 200, 5), (3, 100, 5), (1, 50, 7). Each triplet is interpreted as: (1) minimum number of graph nodes covered by a cell for the pruning condition to be applicable; (2) minimum number of FST states in a cell for the pruning condition to be applicable; (3) pruning threshold used if both (1) and

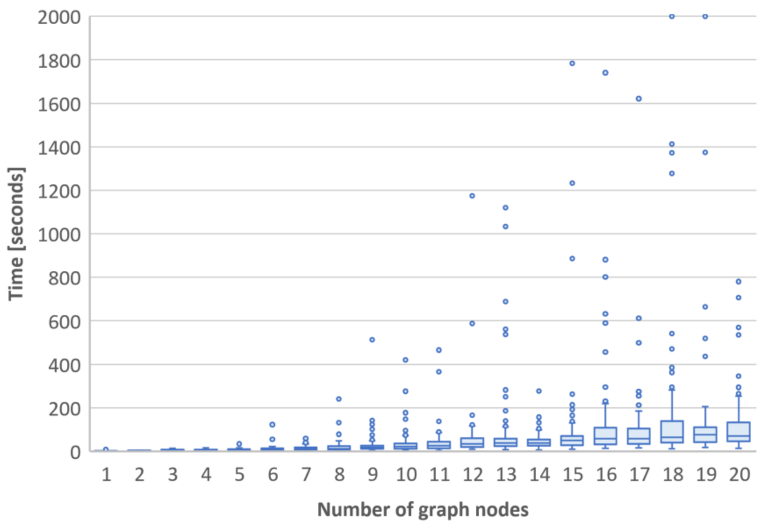


Figure 5.10: Translation task in translation mode runtime including outliers.

(2) are met. Pruning conditions are specified in the same order as presented above; the first applicable condition is used for pruning. The pruning setting was determined experimentally by increasing the severity of pruning until all sentences were translated within specified memory and time constraints.

As described in Section 5.5.2, every time pruning is executed, n-shortest paths are kept regardless of their score; $n = 100$ for both translation and realization task in these experiments. Note that local pruning is not used in alignment mode, where hypothesis space construction is instead constrained to the final pruned FSA of translation mode. This FSA is produced (in translation mode) by keeping 2000 n-shortest paths alongside any path below pruning threshold of 9.

The decoder has limits set on its running time and memory use. In translation mode, it can use up to 40 GB of memory to decode a sentence in up to 2000 seconds on a single CPU (note that language model application and final pruning are excluded from these constraints). The equivalent limits for alignment mode are 60 GB and 1000 seconds.

The experiment results are shown in Figure 5.9. Figure 5.9a shows the running time of the translation task in translation mode. Like similar figures shown in Section 5.7.1, sentences are grouped according to their DMRS graph size and their distributions are shown with box and whisker charts. We can observe that running time increases (reasonably close to) linearly with increased number of graph nodes. As in rule application performance results, dispersion of sentence running time increases with the number of graph nodes representing a sentence.

However, unlike in the equivalent rule application figure (Figure 5.8a), I do not show outliers as they make the resulting figure difficult to interpret. This is because a small number of sentences have a much longer running time than the large majority of them. I show the results with outliers in Figure 5.10 for translation mode of translation task. Note that the sentence running time limit of 2000 seconds was hit in two instances.

Performance between translation and realization tasks is comparable: they are evaluated and trained on the same datasets and use the same pruning settings.[8] However, we can

[8]The English language model was trained on a (significantly) larger monolingual corpus than the German one, but that has a small effect on translation mode running time.

observe a significant difference in running time between Figures 5.9a and 5.9c. The difference stems from the number of applied rules used in each. As seen in the previous section (Figures 5.8b versus 5.8c), there are around twice as many rules per graph node in translation task compared to realization task (800 versus 400), which results in significantly longer decoding times. Note that realization running times could be adjusted to similar levels as in translation task by relaxing the local pruning conditions. As described above, both tasks use the same set of pruning conditions, which were manually tuned to make translation task tractable.

Alignment mode is considerably faster compared to translation mode, which can be observed both for translation and realization tasks. This result may come as a surprise considering that (1) alignment transducers often contain more states and arcs than their translation counterparts, as observed in Section 5.6.1, and (2) the fact that alignment mode is more computationally demanding in HiFST (Allauzen et al., 2014). (1) is explained by alignment mode's constrained hypothesis space generation, which considers far fewer translation hypotheses. (2) is explained by the inherent difference in decoding tasks between HiFST and work presented in this thesis. HiFST decoder (and other Hiero decoders) use CYK to consider many parses of the source sentence string, while graph-to-string decoder considers a single parse of a sentence (although it considers multiple derivations that achieve that parse). This means that graph-to-string decoder considers a smaller set of derivations, trusting that the DMRS graph is a good parse of the source sentence, making alignment mode that computes the derivations less tasking.

Chapter 6

Translation

In this chapter I apply the statistical machine translation approach proposed in this thesis (see Chapters 3, 4, 5) to the problem of machine translation. The approach translates a DMRS graph in the source language to a target language string. A DMRS is a semantic representation of a sentence, modelled as a directed acyclic graph. DMRS was designed to include all semantically related information that can be derived from syntax and morphology. Applying the proposed approach to the machine translation problem therefore follows the trend of using deeper linguistic knowledge in statistical machine translation. It takes advantage of the monolingual knowledge encoded in the high-precision grammar (the ERG) used to obtain DMRS representations from input sentences.

When I first evaluated the performance of the proposed approach on the translation task I found that it did not perform well on real-world data. The system lacked robustness that is inherent in hierarchical phrase-based translation, in particular due to Hiero's glue rules. Consequently, I introduce several types of non-grammar rules, which are not extracted from parallel data but are instead constructed for each input graph as needed during the rule application stage of decoding (see Section 1.5.1). Non-grammar rules include (1) disconnected graph glue rules, which are similar in purpose to Hiero glue rules (Chiang, 2007), (2) node deletion rules, which allow not translating a node if no grammar rules exist for it, (3) carg rules for mapping carg nodes, such as named entities and numbers, directly from source to target side, and (4) mapping rules which allow the same to be done for any source node. With the exception of the last type, non-grammar rules are equally valuable for the realization task (see Chapter 7). I describe the four types of non-grammar rules in detail in Section 6.1.

The approach proposed in this thesis is a large-scale statistical machine translation system (see Section 1.5). Consequently, the proposed approach can be trained on millions of training examples and use the resulting grammar for decoding of real-world inputs on a large-scale. However, a limitation of the current decoder is that it accepts DMRS graphs containing up to 20 nodes. As demonstrated in Section 5.7, this is not an inherent limitation of the approach but rather of its implementation. In lieu of a more efficient implementation, other strategies such as splitting DMRS graphs and decoding them separately, analogous to sentence splitting employed in other SMT systems, can be implemented.

In order to demonstrate its ability to be used as a large-scale statistical machine translation system, I evaluate it on the (large-scale) WMT15 English-German translation task. I compare my approach to a state-of-the-art SMT system, the HiFST implementation of hierarchical phrase-based translation (Blackwood et al., 2016) (see Section 5.1). I describe the set-up of both systems and report on some associated preliminary experiments in Section 6.2. I also describe in more detail the BLEU and METEOR metrics, which are used for evaluating translation quality compared to a reference translation. In addition to evaluation of final system performance, I use BLEU for model parameter tuning (see Section 5.6). Finally, I report and discuss the evaluation results in Section 6.3.

6.1 Non-grammar rules

In Chapter 4 I introduced the rule extraction and grammar construction algorithms which create a translation or a realization grammar. The grammar is subsequently used by the decoder to decode previously unseen DMRS graphs. In practice, however, the decoder encounters graphs which it cannot decode using the extracted grammar: (1) disconnected graphs consisting of several components, and (2) graphs which contain nodes for which no rule exists in the grammar. Consequently, only a part of the input graph can be decoded. In this section, I introduce non-grammar rules which alleviate this problem by enabling the decoder to decode as much of the input graph as possible. These rules significantly improve robustness of the approach presented in this thesis. Unlike the grammar rules, which are extracted from training examples, non-grammar rules are created for each input graph to be decoded at the rule application stage (Section 5.3). They are used both by the translation system described in this chapter and the realization system described in Chapter 7.

The remainder of the section is organized as follows. In Section 6.1.1 I describe glue rules, which enable decoding of disconnected graph components. The glue rules get their name based on their resemblance to glue rules used by Hiero (introduced in Section 4.1). Deletion rules, described in Section 6.1.2, allow decoding of graphs with nodes for which no rules exist in the grammar by deleting the offending node. Instead of deleting the node, carg rules (Section 6.1.3) allow decoding of a node using its carg property. Finally, in Section 6.1.4, I describe mapping rules which allow mapping of source side tokens directly to the target side. They are only applicable to the translation task discussed in this chapter, since source alignment information is not available during realization decoding.

6.1.1 Disconnected graph glue rules

A connected graph (here, as in Section 3.2.3, I refer to the underlying undirected graph of a DMRS graph) is a graph in which there exists a path between every pair of nodes. Conversely, a disconnected graph is a graph that is not connected. A disconnected DMRS graph is caused either by parsing (a bug in the ERG or a consequence of robust parsing settings, see Section 6.2.1) or by one of the DMRS modelling steps (see Section 3.2). Disconnected graphs occur rarely: around 1 graph in every 150 is disconnected in newstest2013 and newstest2014 datasets. A disconnected graph consists of connected components. The example disconnected graph shown in Figure 6.1, for instance, consists of two, marked as $C1$ and $C2$. Disconnected graphs cause problems for the decoder as only the largest component is decoded. The solution I describe in this section creates *glue rules* which join connected components and enable the decoder to decode all of them.

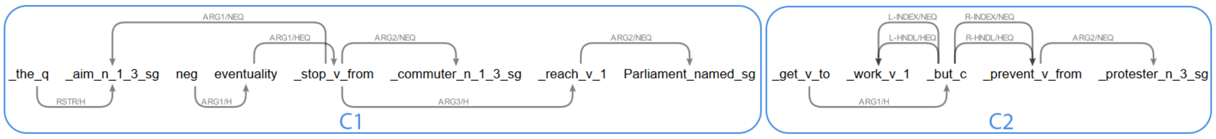


Figure 6.1: An example disconnected DMRS graph presented to the decoder, parsed from "The aim, however, wasn't to stop commuters getting to work but prevent protesters from reaching parliament." The disconnected graph is caused by the DMRS modelling steps.

$$X \rightarrow \langle \text{ X\_0 } \text{ X\_1 } , \text{ X\_0 X\_1 } \rangle$$
$$X \rightarrow \langle \text{ X\_0 } \text{ X\_1 } , \text{ X\_1 X\_0 } \rangle$$

Figure 6.2: Disconnected graph glue rules for the example graph in Figure 6.1.

Recall that the decoder creates the hypothesis space from the cell which covers the largest part of the input graph (Section 5.4.2). Usually this cell is the one covering the entire graph. However, in the case of a disconnected graph, no such cell exists, since no rules exist that would join the two disconnected components of the graph together. Instead, the hypothesis space is created for the disconnected component of the graph that covers the largest part of the input graph, whereas the smaller components of the graph are left undecoded. In the example in Figure 6.1, the C1 component is decoded, while the C2 component is left undecoded.

The solution described in this section creates special glue rules for every disconnected input DMRS graph. A glue rule joining the disconnected components of the graph allows the decoder to decode all disconnected components. On the target side, a glue rule concatenates the target sides of the individual components. Since the target side order of the components is not known, every possible target order (i.e., permutation) is created. Therefore, given a disconnected graph with n components, a set of $n!$ disconnected graph glue rules is created. In practice, only $n = 2$ was observed in newstest2013 and newstest2014 datasets. The two disconnected graph glue rules created for the example in Figure 6.1 are shown in Figure 6.2.

A glue rule allows decoding of disconnected components by placing them in a particular order on the target side. However, this may not create a fluent target side. In the example above, the $C2$ graph component corresponds to the string "getting to work but prevent protesters" in the original sentence. Placing the string (i.e., its realization or translation) before or after the string corresponding to component $C1$ in Figure 6.1 does not yield a fluent output. Instead, the 'correct' placement of $C2$ target string would be inside $C1$ target string. However, since the information where in $C1$ it should be placed is not present in the input graph (since the components are disconnected), it is placed before or after the $C1$ target string.

Disconnected graph glue rules are distinguished from other rules with a **disconnected graph glue rule indicator feature**. Additionally, the rule penalty feature (introduced in Section 4.3.2) is associated with each rule.

6.1.2 Deletion rules

What happens when a node in a graph cannot be decoded because there is no suitable rule in the grammar? The decoder is unable to decode the coverage cell corresponding

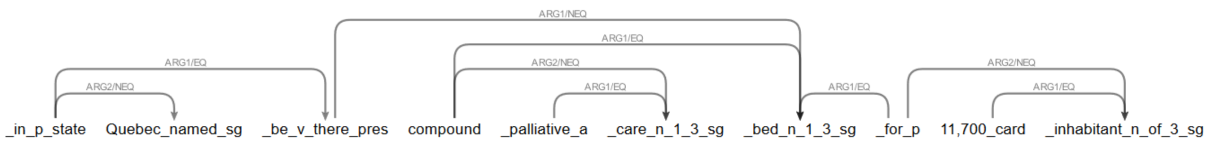
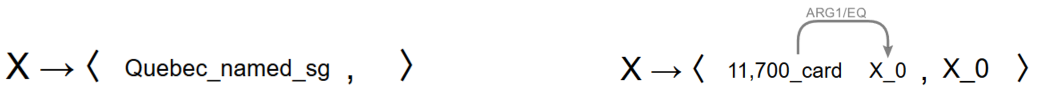


Figure 6.3: An example DMRS input graph to the decoder, parsed from "In Quebec, there are palliative care beds for 11,700 inhabitants."



(a) Terminal deletion rule. (b) Non-terminal deletion rule.

Figure 6.4: Examples of deletion rules constructed for the input DMRS graph in Figure 6.3.

to the node, as well as any coverage cell higher in the hierarchy (cell hierarchy is defined in Section 5.2 and an example is shown in Figure 5.1b). This has severe negative consequences: only the largest subgraph that is not in the hierarchy can be decoded. In this section, I describe *deletion rules* which instead allow deletion of the node for which no rule exists, while enabling other parts of the graph to be decoded fully.

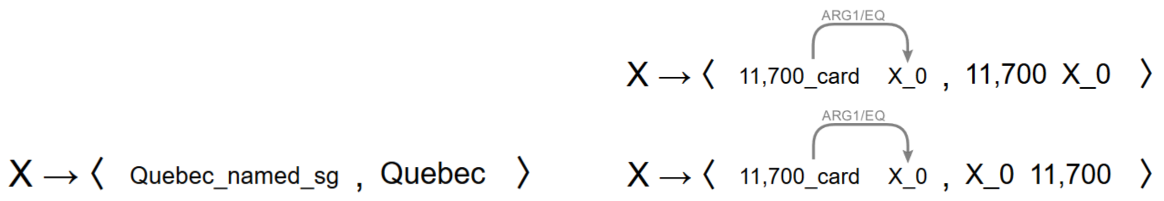
Let's take the DMRS input graph in Figure 6.3 as an example, and assume that no rules exist for decoding nodes `Quebec_named_sg` and `11,700_card`. The cell hierarchy of node `11,700_card` includes nodes `_for_p`, `_be_v_there_pres`, and `_in_p_state`, while the cell hierarchy of `Quebec_named_sg` includes only `_in_p_state`. Consequently, the largest subgraph that can be decoded corresponds to the string "palliative care beds".

Deletion rules are created for every node of the input graph. Two types of deletion rules are created: terminal and non-terminal. A terminal deletion rule is created for a node without any outgoing edges (i.e., without any arguments). An example of a terminal deletion rule for node `Quebec_named_sg` is shown in Figure 6.4a.

In order to allow decoding of node's arguments, a non-terminal deletion rule is created for nodes with outgoing edges (i.e., with arguments). An example non-terminal deletion rule for node `11,700_card` is shown in Figure 6.4b. In the case where a node has more than one argument, a set of $n!$ deletion rules is created in order to allow for all permutations of the n non-terminal tokens on the target side (similar to disconnected graph glue rules described in Section 6.1.1).

A deletion rule (set) is created for every node in the input DMRS graph regardless of whether grammar rules for each node exist. In addition to decoding nodes for which no grammar rules exist, this enables the decoder to delete nodes for which grammar rules do exist. Consequently, deletion rules compete with grammar rules and the decoder must choose between them using its log-linear model and rule features (as described in Section 5.4.3). Similarly to disconnected graph glue rules, deletion rules are distinguished from other rules with a **deletion rule indicator feature**. Additionally, the rule penalty feature (introduced in Section 4.3.2) is associated with each rule.

Returning to our example in Figure 6.3, the decoder provided with the deletion rules (including the two deletion rules shown in Figure 6.4) is now able to decode the subgraph corresponding to the string "In, there are palliative care beds for inhabitants."


X → ⟨ Quebec_named_sg , Quebec ⟩
ARG1/EQ
X → ⟨ 11,700_card X_0 , 11,700 X_0 ⟩
ARG1/EQ
X → ⟨ 11,700_card X_0 , X_0 11,700 ⟩

(a) Terminal carg rule. (b) Non-terminal carg rule set.

Figure 6.5: Examples of carg rules constructed for the input DMRS graph in Figure 6.3.

6.1.3 carg rules

Common numbers and named entities (e.g., ‘one’, ‘David’) are often encountered in the training set. Consequently, rules for decoding them are extracted during grammar extraction. However, there are an infinite number of such predicates which are not present in the training data and which no rules are extracted for. When such predicates occur in the graphs presented to the decoder, they cannot be decoded (instead, they are deleted using deletion rules presented in Section 6.1.2). In this section, I introduce *carg rules* in order to provide rules for such predicates so that they are decoded instead of deleted.

carg rules take advantage of the carg node property (introduced in Section 3.2.1). The carg property stores predicate symbols of numbers, named entities, etc., which are not stored in the lexicon. For example, the input graph in Figure 6.3 contains two carg nodes, `Quebec_named_sg` and `11,700_card`.

A carg rule is created for each carg node of an input graph. Two types of carg rules are created: terminal and non-terminal. A terminal carg rule is created for a carg node without any outgoing edges (i.e., without any arguments). An example of a terminal carg rule for node `Quebec_named_sg` is shown in Figure 6.5a.

As with disconnected graph glue rules (Section 6.1.1) and deletion rules (Section 6.1.2), a set of non-terminal carg rules is created for a carg node with outgoing edges (i.e., with arguments). Each non-terminal carg rule in the set has a different ordering of non-terminal tokens on its target side. An example non-terminal carg rule set for node `11,700_card` is shown in Figure 6.5b.

A carg rule is created for every node with the carg property in the input graph, regardless of whether a grammar rule for it exists. Consequently, like deletion rules, carg rules compete with grammar rules and the decoder must choose between them using its log-linear model and rule features. Two additional features are used to distinguish carg rules from their grammar counterparts:

- **carg terminal rule indicator feature**, equals -1 if the carg rule contains only terminal nodes and tokens (for example, rule in Figure 6.5a), otherwise it equals 0. This feature largely corresponds to rules constructed for `named` carg nodes.

- **carg non-terminal rule indicator feature**, equals -1 if a carg rule contains both terminal and non-terminal nodes and tokens (for example, rules in Figure 6.5b), otherwise it equals 0. This feature largely corresponds to rules constructed for `card` and `ord` carg nodes.

Additionally, word penalty and rule penalty features (introduced in Section 4.3.2) are also associated with each carg rule. Grammar rule features, such as source-to-target and target-to-source probabilities, and rule count indicator, on the other hand, cannot be estimated for carg rules, as carg rules are created independently for each input DMRS graph. Finally, note that carg rules with different target side ordering are discriminated between with the help of a language model.

carg rules often produce the desired decoding of a node, as shown in example rules in Figure 6.5. However, in some cases, carg rules produce a technically correct decoding of a node, but not the desired one. For example, since ERG normalises numbers (e.g., ‘a million’ results in a carg value of ‘1,000,000’), a carg rule decodes a cardinal number node as a sequence of digits instead of words (e.g., ‘1,000,000’ instead of ‘a million’). Similarly, a translation carg rule will decode a named node by preserving its source language form (e.g., ‘United States’ instead of ‘Vereinigten Staaten’). Although still beneficial (and certainly better than deleting the node with a deletion rule), such carg rules may not improve the BLEU score (described in Section 6.2.2), which requires exact n-gram matches.

6.1.4 Mapping rules

Since carg rules (see Section 6.1.3) are limited to nodes with the carg property (e.g., numbers and named entities), I introduce in this section *mapping rules*, which enable transferring the source-side tokens corresponding to every node directly to the target side for the **translation task**.

Mapping rules, like carg rules, rely on source-side information. Instead of using the carg node property, however, they use the source alignment information. Source alignment (introduced in Section 3.3.1) is the alignment between graph nodes and source sentence tokens. Therein lies their limitation to the translation task - source alignment information is not available during decoding for the realization task.

Similarly to carg rules, mapping rules are created for every node in the input graph, with the exception of carg nodes. A mapping rule’s target side consists of the tokens aligned to the particular node, in the order they appear in the source sentence. Identically to carg rules, a set of non-terminal mapping rules needs to be created for each node with outgoing edges (i.e., with arguments) to allow for different non-terminal token ordering on the target side. The full set of translation mapping rules for the example input graph in Figure 6.3 is shown in Figure 6.6.

Mapping rules have a similar feature set as the carg rules: in addition to word and rule penalty features, they are distinguished via two features:

- **Mapping terminal rule indicator feature**, equals -1 if the mapping rule contains only terminal nodes and tokens, otherwise it equals 0.
- **Mapping non-terminal rule indicator feature**, equals -1 if a mapping rule contains both terminal and non-terminal nodes and tokens, otherwise it equals 0.

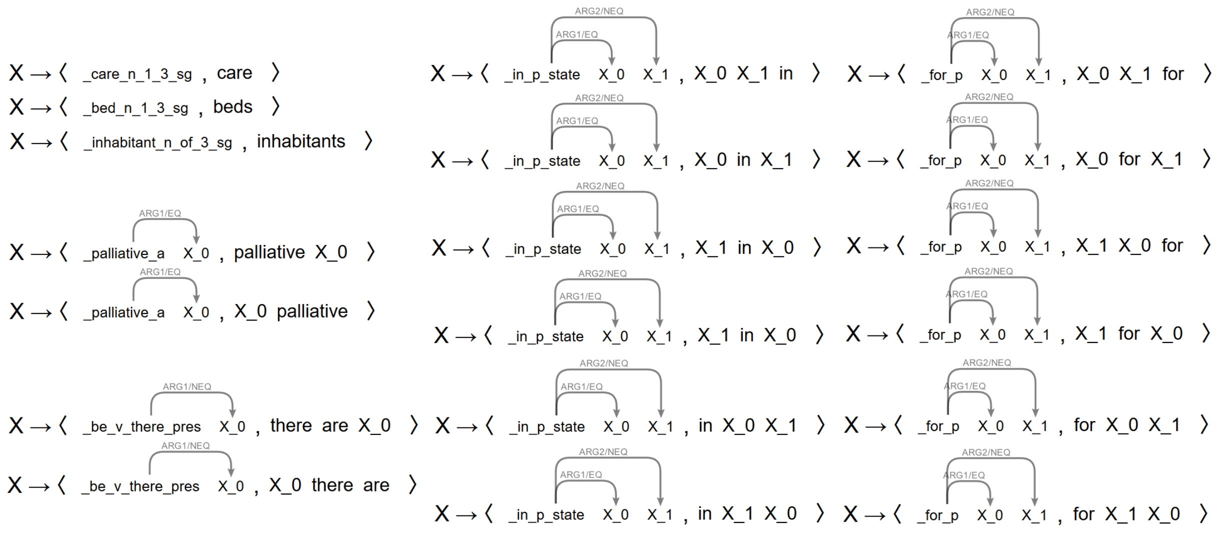

X → ⟨ _care_n_1_3_sg , care ⟩
X → ⟨ _bed_n_1_3_sg , beds ⟩
X → ⟨ _inhabitant_n_of_3_sg , inhabitants ⟩
ARG1/EQ
X → ⟨ _palliative_a X_0 , palliative X_0 ⟩
ARG1/EQ
X → ⟨ _palliative_a X_0 , X_0 palliative ⟩
ARG1/NEQ
X → ⟨ _be_v_there_pres X_0 , there are X_0 ⟩
ARG1/NEQ
X → ⟨ _be_v_there_pres X_0 , X_0 there are ⟩
ARG2/NEQ
ARG1/EQ
X → ⟨ _in_p_state X_0 X_1 , X_0 X_1 in ⟩
X → ⟨ _in_p_state X_0 X_1 , X_0 in X_1 ⟩
X → ⟨ _in_p_state X_0 X_1 , X_1 in X_0 ⟩
X → ⟨ _in_p_state X_0 X_1 , X_1 in X_0 ⟩
X → ⟨ _in_p_state X_0 X_1 , in X_0 X_1 ⟩
X → ⟨ _in_p_state X_0 X_1 , in X_1 X_0 ⟩
X → ⟨ _for_p X_0 X_1 , X_0 X_1 for ⟩
X → ⟨ _for_p X_0 X_1 , X_0 for X_1 ⟩
X → ⟨ _for_p X_0 X_1 , X_1 X_0 for ⟩
X → ⟨ _for_p X_0 X_1 , X_1 for X_0 ⟩
X → ⟨ _for_p X_0 X_1 , for X_0 X_1 ⟩
X → ⟨ _for_p X_0 X_1 , for X_1 X_0 ⟩

Figure 6.6: Set of translation mapping rules created for the example input graph in Figure 6.3.

6.2 Experimental setup

6.2.1 Translation systems

In this section I describe the **HSST** (Hierarchical Statistical Semantic Translation) system evaluated in subsequent sections, and the **HiFST** system used for comparison. In order to make the systems comparable, they are both trained, tuned, and evaluated on the same data, specified and distributed by the WMT15 English-German translation task.[1] I first describe the HSST system, followed by the HiFST system.

The English-German WMT15 training data consists of the parallel corpora from three domains: European parliamentary proceedings, automatically crawled data, and news commentary. I preprocessed and filtered the parallel sentences using the preprocessing steps described in Section 1.5. After preprocessing, the English side of the parallel data was parsed with ERG/ACE (ERG version 1214 and ACE version 0.9.23; Flickinger, 2000; Packard, 2016a). ERG/ACE parsing was conducted with non-default settings: { max-chart-megabytes=7000, max-unpack-megabytes=8000, timeout=15, r='root_informal root_inffrag', tnt-model=wsj.tnt }.

In particular, a shorter timeout is necessary due to the large volume of data. An experiment on a subset of data showed that increasing the timeout beyond 15 seconds did not yield substantial parsing coverage increase (see Table 6.2). The additional root instances allow for more robust parsing, which is necessary to achieve a high parsing coverage on less well-formed data (for instance, crawled data).

The resulting MRS representations were converted to DMRS graphs using the pyDelphin library (version 0.5; Goodman, 2016). The summary of the parallel training corpora is shown in Table 6.1.

In order to tune and evaluate the systems, I used the English-German WMT15 development sets: newstest2013 for tuning and newstest2014 for testing. The sets consists of 3000

[1]WMT15 translation task webpage: http://www.statmt.org/wmt15/translation-task.html

Corpus	Description	Sent. pairs (millions)	Parsed
Europarl v7	European parliamentary proceedings	1.89	80.6%
CommonCrawl	Automatically crawled websites	2.14	81.0%
News Commentary v10	News analysis from Project Syndicate	0.21	85.4%
Combined		4.25	80.8%

Table 6.1: Summary of the parallel training corpora in terms of the number of sentence pairs after preprocessing and the percentage of preprocessed English sentence that were successfully parsed.

Timeout	Timed out %	Parsed %
5 sec	17.9	76.4
10 sec	14.3	80.0
15 sec	7.8	86.2
20 sec	7.0	87.0

Table 6.2: Results of the experiments for the parsing timeout setting conducted on a subset of 1000 English sentences of the News Commentary dataset.

and 3003 manually translated sentence pairs respectively, half of which were translated from English to German, and the other half vice versa.

The two sets were preprocessed in the same way as the training data. The English sides were parsed with relaxed memory and time constraints, since the two sets are much smaller and it is desirable for tuning and test sets to contain as many parsed sentences as possible. The following parse settings were changed for tune and test dataset in comparison to the training data settings: { max-chart-megabytes=14000, max-unpack-megabytes=16000, timeout=120 }. The resulting parsing coverage is 94.3% for newstest2013 and 92.8% for newstest2014.[2]

Since the decoder implementation is not mature (as discussed in Section 5.7), I decided to limit the size of the graphs presented to the decoder (either for tuning or testing) to a maximum of 20 source nodes (and a minimum of 1 node, excluding unparsed sentences).[3] The filtered newstest2013 and newstest2014 datasets contain 1774 and 1574 graph-sentence pairs respectively.

An important aspect of the ERG/ACE setup is the maximum entropy model used for parse ranking (i.e., choosing the order of the parses). In particular, three models trained on data in different domains are available:

- `wsj.mem` trained on Wall Street Journal, treebanked[4] as part of the DeepBank project (Flickinger et al., 2012).

[2]Since the WMT development sets are carefully selected and translated by human translators, it is likely that the increased parsing coverage is in large part a consequence of better-formed text, more so than it is a consequence of the increased parsing settings.

[3]Note that the graph size constraint only applies to graphs used as inputs to the decoder. Graphs of all sizes are used for rule extraction.

[4]Treebanking refers to the manual selection and correction of the top ERG parse.

- `wescience.mem` trained on the WeScience treebank, consisting of Wikipedia articles (Ytrestøl et al., 2008).
- `redwoods.mem` trained on data from a variety of domains, including WeScience and LOGON treebanks. The latter consists of Norwegian tourism brochures (Oepen et al., 2004a).

In an evaluation of parsing performance of the three models on different data domains, Packard (2015) found significant differences between in-domain versus out-of-domain performance. Consequently, I set up a small-scale evaluation to test which of the models is the most appropriate to use for parsing the training, tune, and test datasets. I considered the `wsj.mem` and `redwoods.mem` models, but excluded `wescience.mem`, since none of the data comes from Wikipedia. I compared the 1-best parsed MRS against the gold MRS (obtained via human expert treebanking[5]) on a random sample of 100 sentences from the newstest2013 tuning dataset. Both models achieved good performance. The `redwoods.mem` model matched 43% of the gold MRS representations, compared to the 52% matched by the `wsj.mem` model. Despite these results, I opted to use the `redwoods.mem` model since (1) a large majority of the training data does not consist of news text, (2) it is the more general of the two maximum entropy models, and (3) I have observed that it is important to use a consistent ranking model for all the data, as opposed to using different models on different parts of the data. Therefore, the results can be seen as a reaffirmation that even in the most favourable circumstances (newstest2013 consists of news text), the difference between two models is not too great.

In order to construct the training examples for rule extraction, word alignments between source and target sentences from the training data were extended to alignments between source graphs and target sentences (see Section 3.3). Training data word alignments were obtained by running the MTTK toolkit (Deng and Byrne, 2006), implementing the HMM alignment model, in both directions and unioning the two sets of alignments (see Section 3.3.3). The translation grammar extracted from the training examples consists of 16.5 million rules. The rule extraction and rule application algorithms use $N = 2$ (at most two non-terminals) and $V_{max} = 5$ (at most five source nodes).

The decoder uses local pruning with the following pruning conditions: (1,50,7) (3,100,5) (5,200,5) (8,20,9), and 100 shortest paths (see Sections 5.5.2 and 5.7.2 for explanation). The decoder is limited to 40 GB of memory and 2000 seconds of execution per sentence in the translation mode. The alignment mode uses 5000 top hypotheses from the translation mode and is limited to 40 GB of memory and 1000 seconds of execution per sentence.

The decoder uses a language model as an important feature: a 2-gram FSA language model is used for efficient local pruning during hypothesis space construction (see Section 5.5.2), while a 4-gram language model is applied to the final FSA. The n-gram language models were estimated using the KenLM toolkit with interpolated modified Kneser-Ney smoothing (Chen and Goodman, 1998) on 2.6 billion words of German text specified for the WMT15 translation task. The monolingual German text consists of (1) the German side of WMT15 training data, (2) News Crawl 2007-2014 corpus, and (3) all of German text from the Europarl v7 and News Commentary corpora. The pruning language model was converted from the standard ARPA to an FSA using the OpenGrm library.[6]

[5]I would like to thank Dan Flickinger for the treebanking of the gold dataset

[6]http://www.openfst.org/twiki/bin/view/GRM/NGramLibrary

The **HiFST** system (partly introduced in Section 5.1) is a strong hierarchical phrase-based translation system implementation (Blackwood et al., 2016), trained, tuned, and evaluated on the same data as the HSST system. The two systems also share the same 4-gram language model. The HiFST SCFG consists of shallow-1 rules (Iglesias et al., 2009), which I found to perform best on the English-German translation task (see Section 4.1 for a description of Hiero rule extraction). The systems uses the standard set of features as presented in Iglesias et al. (2009): target language model, source-to-target and target-to-source phrase translation models, source-to-target and target-to-source lexical models, word and rule penalties, number of usages of the glue rule, and three rule count features. Additionally, the system uses provenance features (bidirectional translation and lexical models). Like the HSST system, the HiFST system was tuned using LMERT (Macherey et al., 2008) with the BLEU metric (Papineni et al., 2002) on the tuning set.

6.2.2 BLEU

BLEU is an automatic machine translation evaluation metric introduced by Papineni et al. (2002). It is widely used to measure incremental progress of statistical machine translation systems. The main reasons for its popularity are its inexpensiveness and high correlation with human judgement of translation quality over the entire dataset. Despite its popularity, it is widely acknowledged that BLEU is a flawed metric. Callison-Burch et al. (2006) show that in certain scenarios, BLEU score does not correlate well with human judgement of translation quality and that the metric should not be used for comparisons of systems with radically different approaches to machine translation. I use the BLEU metric for tuning the approach proposed in this thesis (see Section 5.6) and to evaluate performance of the translation systems discussed in this chapter and the realization systems in Chapter 7. In the remainder of this section I describe how the BLEU score is computed.

The BLEU metric measures the closeness of a translation hypothesis to a reference translation using N-gram precision:

$$\text{BLEU} = \text{BP} \cdot \exp\left(\frac{1}{N}\sum_{n=1}^{N} \log p_n\right) \tag{6.1}$$

where p_n is the clipped N-gram precision, computed by dividing the number of N-gram matches by the total number of n-grams in the translation, clipped to the maximum number of occurrences of the N-gram in the reference; N is the highest order of N-grams used in the computation of the BLEU score (frequently, $N = 4$). $\frac{1}{N}\sum_{n=1}^{N} \log p_n$ is therefore a geometric mean of N N-gram precisions.

BP refers to the brevity penalty, which penalizes short translations. Short translations receive high N-gram precision scores, as it is easier to achieve high precision for a small number of N-grams compared to a large number of N-grams. The brevity penalty is computed using the following formula:

$$\text{BP} = \begin{cases} 1 & \text{if len(trans)} > \text{len(ref)} \\ e^{(1-\text{len(ref)}/\text{len(trans)})} & \text{otherwise} \end{cases} \tag{6.2}$$

The BLEU score is commonly computed over the entire dataset and can be easily extended to use multiple reference translations in order to get a more reliable relative score. I use the NIST's mteval-v13[7] script to compute BLEU scores in this thesis.

6.2.3 METEOR

METEOR is an automatic machine translation evaluation metric introduced by Banerjee and Lavie (2005) which addresses some of the weaknesses of the BLEU metric. A major weakness of the BLEU metric (see Section 6.2.2) is that it does not measure recall directly. It indirectly compensates for it via the brevity penalty. The METEOR metric addresses this problem by computing both (unigram) precision and recall based on an alignment between the hypothesis and translation.

The METEOR score of a single pair of hypothesis and reference translation is computed based on generalized unigram matches between them. A unigram match between a hypothesis and a reference occurs if two unigrams share the same surface or stemmed form. Additionally, WordNet synonyms are considered. Based on the unigram matches, alignments between the hypothesis and reference translation are created. An alignment is a mapping between unigrams such that every unigram in a translation maps to zero or one unigram in the other translation. For each alignment, a score is computed as a combination of unigram-precision, unigram-recall, and fragmentation penalty. Unigram precision is computed as a ratio of mapped hypothesis unigrams to all *hypothesis* unigrams, while unigram recall is the ratio of mapped hypothesis unigrams to all *reference* unigrams. From unigram precision and unigram recall, the F_{mean} score is computed:

$$F_{mean} = \frac{10 \cdot P \cdot R}{9 \cdot P + R} \tag{6.3}$$

Fragmentation penalty accounts for gaps and differences in the word order and is computed as follows:

$$\text{Pen} = 0.5 \cdot \left(\frac{\#chunks}{\#matched\_unigrams}\right)^3 \tag{6.4}$$

where a chunk comprises of adjacent unigram matches. The score of an alignment is computed as:

$$\text{score} = (1 - \text{Pen}) \cdot F_{mean} \tag{6.5}$$

The METEOR score of a single pair of hypothesis and reference translation is chosen as the maximal scoring alignment score between the two. The overall system METEOR is computed from aggregate statistics accumulated over the entire test set.

Several generalizations and additions were introduced in the subsequent versions of the METEOR metric. In particular, values such as 0.5 and 3 in penalty computation and the balance between precision and recall in the F_{mean} were parametrized and are tuned to maximize correlation with human judgements for each target language (Lavie and Agarwal, 2007). Additionally, unigram precision and recall distinguish between content

[7] Available from ftp://jaguar.ncsl.nist.gov/mt/resources/mteval-v13.pl

	newstest2013			newstest2014		
System	BLEU	BLEU-cased	METEOR	BLEU	BLEU-cased	METEOR
HSST	13.1	12.5	31.8	12.7	12.0	33.2
HiFST	20.2	19.2	41.0	20.0	19.0	42.5

Table 6.3: Translation performance of the HSST and HiFST systems on newstest2013 and newstest2014 datasets (tune and test set, respectively). Performance of the systems is characterized in terms of case-insensitive and case-sensitive BLEU scores and METEOR score.

and function words. In the following section, I use METEOR version 1.5 (Denkowski and Lavie, 2014).[8]

6.3 Translation evaluation

In this section I present the translation evaluation results of the proposed approach. The automatic evaluation metric results are summarized in Table 6.3. The HSST system is outperformed by the HiFST system according to both BLEU and METEOR metrics. The difference between the systems is smaller according to the METEOR metric. As expected, both system perform worse according to the case-sensitive BLEU score (BLEU-cased), although the decrease in performance is somewhat smaller for the HSST system than for the HiFST system (-0.6 and -0.7 BLEU points versus -1.0). Both systems also perform slightly worse on the test set than they do on the tuning set in terms of BLEU score. This is expected, since the systems were tuned to improve the tuning set BLEU score. On the other hand, the METEOR scores are higher for both systems on the test set.

In addition to the automatic translation quality evaluation, I conducted a small-scale human evaluation of the translation outputs, similar to the evaluation described in Bond et al. (2011). The evaluation was conducted on a random set of 100 sentences from the newstest2014 dataset. Two native German speakers[9] were asked to compare the HSST and HiFST translations to a German reference sentence and decide which translation they preferred (giving it a score of 1.0). The competing translation outputs were presented in randomised order so that the evaluators did not know which system each translation came from. A third option[10] allowed the evaluators to rank the systems equally (giving each a score of 0.5). Under these criteria, the HSST system achieved a combined score of 31.25, while HiFST scored 68.75. The human evaluation therefore confirmed the results of the automatic evaluation using BLEU and METEOR metrics. The full evaluation results for the 100 sentences are reported in Appendix A.3.

Since both systems make various translation errors, it is informative to investigate the types of errors they make and compare the systems in that respect. Such an investigation requires some linguistic intuition in order to recognize different types of errors.

[8]Available at https://www.cs.cmu.edu/~alavie/METEOR/. I used the German ranking parameters, tuned on the WMT human binary system rankings. Additionally, normalization and matching with *exact*, *stem*, and *paraphrase* were used.

[9]Neither of the evaluators had a (computational) linguistics background.

[10]The third option was 'Unentschieden' (or 'Equal'). Special thanks goes to Alex Kuhlne for his help with translating the instructions into German and choosing instructive examples.

Consequently, in addition to a linguistically-uninformed evaluation, I conducted a limited informal linguistically-informed investigation into the performance of the HSST system and its differences compared to the HiFST system, with native and non-native German speakers who are familiar with linguistics. I report the findings of the investigation in Section 6.3.1. I discuss the reasons for performance differences and potential future improvements in Section 6.3.2.

6.3.1 Analysis

In this section I describe some initial findings of the manual output investigation of translation outputs. The description is aided by example translations shown in Figure 6.7. Additional example translations are shown in Appendix A.2.

I start by showcasing the use of non-grammar rules in translation examples (discussed in Section 6.1). In the first example, the HSST system makes use of the carg rules (see Section 6.1.3) to translate *European* and *ESO* directly from the source sentence in English to the target sentence in German. In the reference sentence, *European* is not used in its source form, whereas *ESO* is. The first part of the phrase, *Southern*, is not a carg, so the HSST system translates it as *südlichen*. In comparison, HiFST translates the entire phrase in its English form. In the second example, the HSST system does not translate *South* (in *South America*), employing a deletion rule (see Section 6.1.2).[11]

In both examples, we can observe that the HSST translation is missing end punctuation. This is a consequence of DMRS graphs not directly representing punctuation, meaning that the punctuation tokens are not aligned to any graph nodes. Since the rule extraction algorithm does not allow unaligned tokens at the either end of the target token sequence, end punctuation is omitted by the HSST system (see Section 3.3.1 for more details).

In our investigation, we observed that HiFST tends to omit the verb in translation more frequently than HSST. For instance, the HSST translates the verb in example 3 as *führt* (although in incorrect tense), whereas HiFST only translates the auxiliary *hat*. In example 4, HSST correctly translates *had* as *hatten*, whereas HiFST omits it altogether.[12] Finally, in example 5, HiFST again translates auxiliary *habe* but not the main verb *went*, whereas the HSST system translates the verb correctly.[13]

These three examples also show that BLEU is a flawed translation evaluation metric. Omitting the verb is a severe translation mistake, but is not treated differently than, for instance, omitting a function word, since BLEU makes no distinction between the types of words in a matching n-gram (see Section 6.2.2). METEOR does make a distinction between matching function and content words, but does not distinguish between adjectives, nouns, and verbs.

Example 6 demonstrates several problems occurring in HSST translations (although atypically many in a single translation): incorrect gender agreement (*Vater* and *die*), incorrect lexical choice (verb *vertrieben wurden*), and inserted word (*auch*).[14] In example 7, parts

[11]Neither system translates the second example particularly well (e.g., poor lexical choice, and wrong tense in the HSST translation).

[12]Additionally, HSST does not translate *Communist*, and both translations have poor agreement.

[13]But incorrectly inserts *Noch* at the beginning of the sentence.

[14]Additionally, *coffin of my Oxford career* is translated as *coffin of Oxford, my career.*

1. *View of the European Southern Observatory (ESO) in the Chilean Andes.*
   Blick auf das Observatorium der Europäischen Sternwarte (ESO) in den Chilenischen Anden.
   H Blick auf die südlichen European Observatory, ESO in den chilenischen Anden
   B Angesichts des European Southern Observatory (ESO) in den chilenischen Anden.

2. *Those who speak with authority never commit to anything, whereas we South Americans sign everything.*
   Die, die lehren, verpflichten sich nie zu etwas, und im Gegenzug dazu unterzeichnen wir die Südamerikaner immer alles.
   H Diejenigen, die sich mit der Kompetenz habe noch nie etwas begangen, während wir, die Amerikaner haben alles unterzeichnet
   B Diejenigen, die sich mit der Behörde nie zu irgendetwas verpflichten, während wir im Süden die Amerikaner alles unterschreiben.

3. *The Internet has caused a boom in these speculations.*
   Das Internet hat diese Spekulationen erhöht.
   H Das Internet führt zu einem Boom in diesen Spekulationen
   B Das Internet hat zu einem Boom in diesen Spekulationen.

4. *Even my Communist friends always had impeccable manners.*
   Selbst meine kommunistischen Freunde legten stets einwandfreie Manieren an den Tag.
   H Selbst meine Freunde hatten immer tadellosen Manieren
   B Sogar meine kommunistischen Freunden immer tadellosen Manieren.

5. *I never went back.*
   Ich kehrte nie zurück.
   H Noch nie ging ich zurück
   B Ich habe nie wieder.

6. *It was my father who drove the nail into the coffin of my Oxford career.*
   Es war mein Vater, der den letzten Nagel in den Sarg meiner Oxford-Karriere schlug.
   H Mein Vater war es auch, die den Nagel in den Sarg von Oxford, meine Karriere vertrieben wurden
   B Es war mein Vater, fuhr den Nagel in den Sarg meiner Karriere.

7. *Honestly, there's only one reason I'd stop, and that's my little girl.*
   Ehrlich gesagt gibt es nur einen Grund, warum ich aufhören würde, und das ist mein kleines Mädchen.
   H Ehrlich gesagt, es würde ich, nur ein Grund zu stoppen, und das ist mein kleines Mädchen
   B Ehrlich, es gibt nur einen Grund, ich würde aufhören, und das ist mein kleines Mädchen.

Figure 6.7: Example HSST and HiFST translations from the newstest2013 dataset. For each example I show the original English sentence and its reference translation (in italics), followed by the HSST translation (*H*) and HiFST translation (*B*).

1. *He loves well-educated people who come from good backgrounds, and they love him.*

   Er mag Leute mit guter Erziehung und den höchsten Titeln, und sie lieben ihn.

   H Er liebe es, dass gut gebildete Menschen, die aus guten Verhältnissen, und sie lieben ihn

2. *However, the European Central Bank (ECB) took an interest in it in a report on virtual currencies published in October.*

   Dennoch hat die Europäische Zentralbank (EZB) in einem im Oktober veröffentlichten Bericht über virtuelle Währungen Interesse hierfür gezeigt.

   H Aber die Europäische Zentralbank EZB hat ein Interesse daran, dass es in einem Bericht über virtuelle Währungen, die im Oktober veröffentlicht worden

3. *Jorge says he knows the 35 friends he has on FB and his nine followers on Twitter.*

   Jorge versichert, 35 Freunde zu kennen, die er in FB hat und neun Folger in Twitter.

   H Jorge sagt, dass er weiß, dass die 35 Freunde, die er auf FB und neun seiner Anhänger auf Twitter

4. *This season, you have taken on a new stature with PSG.*

   In dieser Saison haben Sie mit PSG ein neues Ausmaß angenommen.

   H Sie hat sich eine neue Statur mit PSG in dieser Saison

5. *Who came up with this idea?*

   Von wem stammt diese Idee?.

   H Wer kann denn wirklich mit dieser Idee

Figure 6.8: HSST translation examples with inserted filler words or sentence structures. For each example I show the original English sentence and its reference translation (in italics), followed by the HSST translation (*H*).

of the HSST translation are correct,[15] however the middle of the sentence (*es würde ich, nur ein Grund zu stoppen*) could be described as a 'word salad'.

A recurrent problem observed with HSST translations is insertion of filler words or sentence structures that are not warranted by the source sentence or its analysis. I show additional example translations with such problems in Figure 6.8. A frequently inserted sentence structure is the relative clause (starting with *dass* in examples 1, 2, and 3). Individual filler words, such as *wirklich* and reflexive *sich*, are also inserted occasionally (examples 4 and 5). I discuss potential reasons for these problems in Section 6.3.2.

In summary, the manual analysis confirmed that the HSST system performs worse than HiFST, although perhaps not to the degree indicated by the evaluation metric scores. A potential strength of the HSST system in comparison to HiFST is translation of verbs. In

[15]In particular, *Ehrlich gesagt* and *und das ist mein kleines Mädchen.*

the following section I discuss some of the reasons for the difference in system performance.

6.3.2 Discussion

The manual investigation of translation outputs reported in Section 6.3.1 indicated that insertion of relative clauses and words such as ‘sich’ into translations when not warranted by the source sentence is a recurrent problem of the HSST system. I hypothesize that this behaviour originates in the inability of the HSST system to translate certain parts of the input graphs (i.e., due to missing grammar rules) and instead being forced to use deletion rules (see Section 6.1.2). Since the system is tuned to optimize the BLEU score on the tuning dataset, it is severely penalized for producing short outputs via the brevity penalty (see Section 6.2.2). Consequently, it is likely that the parameter tuning results in the system compensating for short outputs by producing unnecessarily long outputs when it can, in the form of relative clauses and inserted words. This affects the translation quality of all sentences, not only the sentences for which the system is missing grammar rules.

This observation led me to work on improving the capabilities of the system to translate (parts of) input DMRS graphs that it previously could not. Notable parts of the approach presented in this thesis are a result of these efforts, including graph cycle removal (see Section 3.2.3), disconnected graph glue rules (see Section 6.1.1), and 3-arg deletion rules. The latter are not described explicitly, but rather form a subset of deletion rules (see Section 6.1.2) which enable translation of predicates with more than two arguments via deletion. These additions helped reduce the frequency of the insertion problems and improve the quality of the system in terms of BLEU score. Despite the reduced frequency, the insertion problems persisted as shown in Section 6.3.1.

However, when investigating the outputs of the *realization* system (reported in Chapter 7), no such problems were observed. Despite the inherent difference in the two tasks, a comparison between the two systems is intriguing. The translation and realization systems are based on the identical approach proposed in this thesis. However, the realization system performs significantly better in terms of producing grammatical and fluent outputs that preserve the meaning of the DMRS meaning representation (see Section 7.3) compared to the translation system, when trained, tuned, and evaluated on data of equal size. I discuss the differences between the translation and realization systems below in order to further elucidate the translation system performance.

The DMRS representation is not an interlingua, but is instead a language-specific abstraction from a surface sentence. Consequently, there is a close to one-to-one correspondence between English surface tokens and English DMRS predicates. The correspondence between English DMRS predicates and German surface tokens, on the other hand, is not as close. In other words, there are significantly more ways of translating an English predicate into German, than there are ways of realizing an English predicate. The difficulty of the English-German translation task in particular is further exacerbated by the rich German morphology, which results in even more possible ways of translating a single English predicate (i.e., with different suffixes, such as ‘-e’, ‘-en’, ‘-em’, ‘-es’). A simplified conclusion is that the translation task is more difficult that the realization task.

However, one might expect that as a result, the translation grammar would be significantly larger than the realization grammar. As we observed in Section 4.4, the exact opposite is true: the translation grammar is significantly smaller than the realization grammar at all training set sizes (29 versus 16 million rules, extracted from the full training dataset), despite using the identical grammar extraction procedure. The difference in rule extraction can only be attributed to the alignments between source-side nodes and target-side tokens. Since English predicates do not correspond to German tokens as well as to English tokens, the alignments between them are less uniform. Additionally, the reordering between the two languages means that tokens that are contiguous in English can appear at different parts of the sentence in German. Consequently, the terminal and non-terminal constraints (see Section 4.2.3) are more constraining for the translation task than they are for the realization task, which results in fewer rules being extracted.

The translation system could potentially require more training data to achieve better performance. However, preliminary experiments on smaller training set sizes using an earlier version of the system suggested that this is not the case. Although increasing the training set size does improve translation performance (e.g., +1.0 BLEU point going from 1 to 2 million training examples), the increases provide diminishing returns (+0.5 BLEU point going from 2 to 4.25 million training examples).

The rule extraction problem is further exacerbated due to noise in translation alignments. Whereas realization alignments between DMRS nodes and English tokens are obtained via the parser and heuristic methods (see Section 3.3.1), they need to be extended to German tokens for the translation task. This is achieved via a noisy statistical unsupervised approach (see Section 3.3.3), which introduces significant errors into the translation training data. Neubig and Duh (2014) demonstrated that the alignment quality has a significant effect on the performance of a syntax-based tree-to-string translation approach, much more so than on the (hierarchical) phrase-based systems, which are more resilient. It is likely that the graph-to-string approach presented in this thesis is similarly affected by the translation alignment quality.

Preliminary experiments with an alternative statistical word aligner, GIZA++, did not show promising improvements in terms of translation quality. Following Neubig and Duh (2014), I also experimented with the Nile toolkit (Riesa and Marcu, 2010), a supervised alignment approach which takes advantage of source and target syntax information. However, the Nile toolkit proved to be impractical to run on the training set sizes used in this thesis.

Based on observations from manual investigation (see Section 6.3.1), there is potential for my approach and HiFST to complement each other in their strengths. Although I do not report it in this thesis, I attempted several low-level integration strategies between the two systems. One type of integration combined HiFST and HSST FSAs of cells which covered the same spans. Such integration has potential to alleviate the problem of missing HSST translation rules and complement them with HiFST rules. Unfortunately, the attempted integration strategies did not yield the desired results in preliminary experiments and would require further work.

In summary, the HSST system does not perform as well as HiFST due to coverage problems stemming from rule extraction. Nonetheless, promising avenues for improvements exist, including improving translation alignment quality and integrating the proposed approach with a complementary SMT system such as HiFST.

Chapter 7

Realization

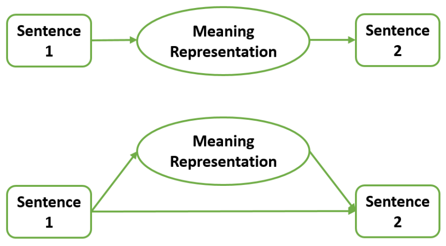
Realization, introduced in Chapter 1, is most often interpreted as the task of creating a fluent sentence from a meaning representation. This interpretation of the realization task assumes that the meaning representation is perfect. It corresponds to the classical task of tactical generation. The main criterion for this interpretation of realization is preservation of meaning as specified by the meaning representation. Additional criteria are grammaticality and fluency of the output as well as coverage and robustness.

In contrast, the *regeneration* interpretation of the realization task does not assume that the meaning representation is perfect. Instead, it accepts that the meaning representation is potentially a flawed representation of the original sentence (or another object) because it was obtained via an imperfect process (e.g., automatic parsing). This interpretation is more application-orientated than the tactical generation interpretation, since it perceives realization in the extended context in which the meaning representation was obtained as opposed to an isolated task (i.e., the pipeline view). The difference is illustrated in Figure 7.1a.

The main criterion of realization under the regeneration interpretation is no longer meaning preservation as specified by the meaning representation, but meaning preservation as intended by the original sentence. Whereas in the first interpretation, not realizing a flawed or broken representation may be seen as acceptable, that is not the case under the second interpretation. Consequently, the coverage and robustness criteria are paramount in addition to meaning preservation. Grammaticality and fluency of the output remain of equal importance.

The meaning representations considered in this thesis are (D)MRS representations (introduced in Section 3.1). The established approach to realization of MRS (and since they are inter-convertible, DMRS) representations uses chart generation against a grammar, as presented by Carroll et al. (1999). Consequently, both parsing and realization rely on the same grammar (in this thesis, the ERG). In addition to chart generation, the realizer uses a maximum entropy model for realization reranking (Velldal and Oepen, 2005; Velldal, 2007).

The reliance on a grammar is both a strength and a weakness of the established approach to realization. The realizer tends to produce high quality grammatical realizations for representations that were created by a parser using the same grammar. In fact, the approach is guaranteed to be meaning preserving (with respect to the meaning representation) and to output only grammatical sentences, while output fluency is addressed with realization



(a) The tactical generation interpretation (top) and the regeneration interpretation (bottom) of the realization task.

	Parsing	Realization
Precision	ERG/ACE	ERG/ACE
Robustness	Robust parsing with PCFG	**Proposed approach**

(b) Relationship between ERG/ACE parsing and realization approaches compared to robust parsing of Packard and Flickinger (2017) and robust realization proposed in this thesis.

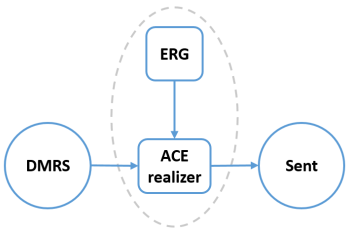
Figure 7.1: Realization task interpretation (a) and relationship between parsing and realization approaches (b).

reranking. This makes the established approach to MRS realization particularly well-suited to the tactical generation interpretation of the realization task. However, when presented with meaning representations that have been created by an external system or that have been augmented or modified after they have been parsed, the realization approach has proved to be brittle. Instead of degraded performance, the somewhat flawed or broken representations result in the failure of the realizer to produce any output. In many such applications, for instance in the transfer-based machine translation system LOGON (Lønning et al., 2004), much effort has been spent on ensuring that the resulting representations are such that they can be realized. The lack of robustness makes the established approach to MRS realization on its own less well-suited for the regeneration interpretation of realization.

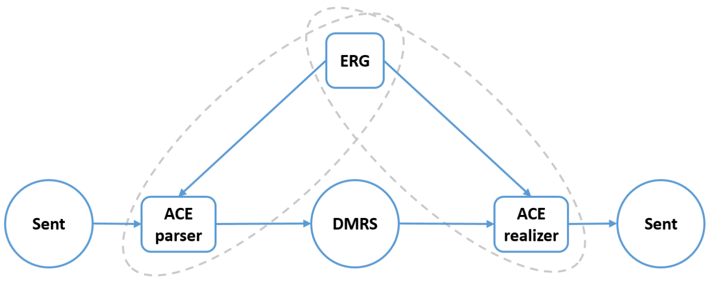
The machine translation approach to realization presented in this thesis provides a complementary set of strengths and weaknesses. It does not hope to surpass the output quality (in terms of grammaticality and fluency) of the established realizer, although an objective is to come as close as possible. Instead, it provides a robust alternative that is capable of producing a realization even when the input representations are flawed. Consequently, the approach is well-suited for the regeneration interpretation of the realization task.[1]

Robustness is a weakness of both ERG/ACE realization and parsing approaches. In many DELPH-IN applications, a common strategy is to back-off to a robust statistical approach when the original system fails to produce an output. The downside of such a strategy is that the statistical approaches are decoupled from the original system and therefore do not take advantage of any ERG knowledge. Recently, a robust parsing approach was introduced by Packard and Flickinger (2017) following the work of Zhang and Krieger (2011). The robust parsing approach is capable of achieving much higher parsing coverage than the established approach, including parsing of ungrammatical inputs. The approach is based on a PCFG estimated on ERG parse trees and is therefore more tightly coupled with the existing parsing approach compared to the standalone statistical back-off approaches. So far, no such solution existed for robust realization. I argue that the machine translation approach presented in this thesis is the realization equivalent to robust parsing. The robust realization approach shares many objectives with the robust parsing

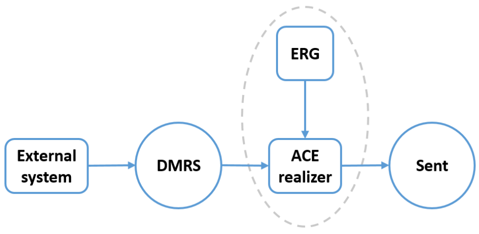
[1]A drawback of the decoder is its constraint of decoding graphs with at most 20 nodes. However, the limitation is not inherent to the approach, but to the current implementation of it, as demonstrated in Section 5.7.



(a) The realization setting in which the DMRS is assumed to be correct.



(b) The realization setting in which the DMRS is created by the parser.



(c) The realization setting in which the DMRS is created by an external system.

Figure 7.2: The three realization settings considered in the evaluation.

approach: (1) it is capable of realizing inputs that are imperfect; (2) given flawed inputs, it can produce outputs of degraded quality instead of failing to produce an output at all; and (3) it is trained on representations produced by the ERG and is therefore more tightly coupled with the existing realization approach. The relationship between the approaches is illustrated in Figure 7.1b.

In this chapter, I evaluate and compare the machine translation approach to realization with the established approach under both interpretations of the realization task. Since the interpretation of the realization task affects how the systems should be evaluated, I define three realization scenarios. The first scenario corresponds to the tactical generation interpretation in which the representations are assumed to be correct. The second and third scenarios both correspond to the regeneration interpretation but differ on how the meaning representation is obtained. The second scenario considers meaning representations obtained via parsing, whereas the third scenario considers representations created or modified via an external system (e.g., a transfer-based machine translation system). The difference between the two scenarios is significant for (D)MRS representations because producing DMRS graphs with an external system can potentially result in DMRS graphs which are not only flawed (as when produced via parsing), but also not well-formed.[2] I

[2]Well-formedness of (D)MRS representations is not used as a precise term in this chapter, but rather as a conflation of several types of (D)MRS well-formedness.

refer to such representations as *non-ERG*. The three different scenarios are illustrated in Figure 7.2.

This chapter is structured as follows. In Section 7.1.1 I give details on the four systems considered in this chapter. I refer to my approach as the HSSR system, and introduce two established realization systems, ERG/ACE and ERG/ACE++, as well as a combined approach ERG/ACE++HSSR. I describe the evaluation metrics used in this chapter in Section 7.1.2.

I compare my approach to the standard approach to realization in each of the three scenarios outlined above. In order to evaluate the realization systems in the first scenario (the meaning representations are considered perfect), I obtained manually-curated gold representations. I present the experiment, results, and subsequent analysis in Section 7.2. In Section 7.3, I evaluate the realization systems in the second realization scenario, realizing representations obtained from a parser. Finally, in Section 7.4, I consider the third scenario, in which representations are created or modified by an external system. Initially, I demonstrate the ability of the HSSR system to realize heavily modified and compressed DMRS graphs, output by a simplification application. I continue by comparing and analysing the performance of the established approach against the HSSR system on representations produced as a result of robust parsing with a PCFG. Finally, I use the HSSR system as the realization component of a transfer-based machine translation system and demonstrate its robustness compared to the established approach.

7.1 Experimental setup

7.1.1 Realization systems

HSSR or Hierarchical Statistical Semantic Realization system uses the same implementation as the HSST system introduced in Section 6.2.1 (recall that the system overview is given in Section 1.5).[3] The HSSR system is trained on the English side of the WMT15 English-German data introduced in Section 6.2.1. The two systems share the parsed MRS representations of the English sentences (refer to Section 6.2.1 for ACE parsing settings). The system was trained on lower-case English target strings in order to help reduce the grammar and language model sparsity. Similarly, the filtered English sides of newstest2013 and newstest2014 datasets introduced in Section 6.2.1 were used for tuning and testing the HSSR system, respectively. As with translation, both datasets were filtered to contain graphs with at most 20 nodes. Additional datasets used in the experiments are described in their respective sections.

The bigram pruning FSA and the final four-gram language model were estimated using the same settings as with the HSST system (refer to Section 6.2.1) on 8.6 billion words of English text, consisting of (1) the English side of WMT15 training data, (2) News Crawl 2007-2014 corpus, (3) News Discussion 2015, and (4) all parts of the 5th edition of the

[3]The cycle removal step of graph modelling differs slightly between the two systems (the translation system can use source sentence token distance while the realization system cannot; see Section 3.2.3).

English Gigaword.[4]

The realization grammar extracted from the training examples consists of 28.6 million rules. The rule extraction and rule application algorithms use $N = 2$ (at most two non-terminals) and $V_{max} = 5$ (at most five source nodes). The decoder uses local pruning with the following pruning conditions: (1,100,7) (3,200,5) (7,500,7) (8,100,9), and 200 shortest paths (see Sections 5.5.2 and 5.7.2 for explanation). As with the HSST system, the decoder is limited to 40 GB of memory and 2000 seconds of execution per sentence in the translation mode. The alignment mode uses 5000 top hypotheses from the translation mode and is limited to 40 GB of memory and 1000 seconds of execution per sentence.

ERG/ACE is the established realization system for English as available out of the box. It uses the English Resource Grammar version 1214 and the ACE realizer version 0.9.23. The system uses default memory and timeout settings (experiments on a subset of data did not show improvement in realization quality or coverage with increased settings).

ERG/ACE++ is an improved version of the ERG/ACE system with a simple approach to unknown word handling. Unknown words are words that do not appear in the ERG lexicon. They are nonetheless parsed by the ERG/ACE parser and assigned a Penn treebank part-of-speech tag (e.g., NNS, JJ). However, the realization part of the ERG/ACE system is currently not able to realize such predicates, which leads to a failure to realize a meaning representation. ERG/ACE++ bypasses this problem by introducing a preprocessing step that converts the unknown predicate to a known predicate based on its part-of-speech tag (for example, `_fermions/NNS_u_unknown` is converted to `_cat_n_1` and `_curative/JJ_u_unknown` to `_sweet_a_to`). A dictionary of up to five different substitute predicates for each relevant part-of-speech tag was manually specified. Subsequently, the preprocessed MRS representation is realized by the ERG/ACE system. Finally, in a postprocessing step, the known realizations of the substitute predicates are replaced with the original unknown word forms (for example, `_cat_n_1` given NNS part-of-speech tag is realized as *cats* and subsequently replaced with *fermions*).[5]

ERG/ACE++HSSR is a simple system combination of the ERG/ACE++ system and the HSSR system. Namely, the HSSR system is used when the ERG/ACE++ system fails to produce an output.

7.1.2 Evaluation measures

BLEU is a machine translation quality metric introduced by Papineni et al. (2002). I described the BLEU metric in Section 6.2.2. Apart from being ubiquitous in (especially statistical) machine translation, BLEU is often used to evaluate the output of realization systems (Belz et al., 2011). Espinosa et al. (2010) investigated the use of various automatic evaluation metrics to measure the quality of realization output and found that BLEU correlates moderately well with human judgment of adequacy and fluency. They

[4]The 5th edition of the English Gigaword corpus (LDC2011T07) consists of newswire text from seven international sources: Agence France-Presse, English Service (afp), Associated Press Worldstream, English Service (apw), Central News Agency of Taiwan, English Service (cna), Los Angeles Times/Washington Post Newswire Service (ltw), New York Times Newswire Service (nyt), Washington Post/Bloomberg Newswire Service (wpb), and Xinhua News Agency, English Service (xin).

[5]Credit for the solution on the plaintext version of this problem goes to Ewa Muszyńska, although solutions in similar spirit may have a long history.

concluded that BLEU is a useful metric for measuring incremental progress of a realization system, but needs to be complemented with other methods when comparing different realization systems. In particular, BLEU does not measure whether the realizations are meaning-preserving and grammatical. Because of this, I introduce two additional metrics to evaluate the quality of the HSSR system (described below).

ERG/ACE match metric compares the realization outputs of the HSSR system against the ERG/ACE++ system. The metric is defined as the percentage of matched realized sentences out of those that were realized by the ERG/ACE++ system in a given dataset. For a match to occur between a pair of realized sentences, an exact string match is required after the following text normalizations: (1) normalizing whitespace to single spaces, (2) removing case information, and (3) removing punctuation. The ERG/ACE match metric can be generalized from comparing the top realization outputs of the two systems, to comparing the n_1-best HSSR realizations against the n_2-best ERG/ACE++ realizations. In the general case, a match occurs if at least one pair of realizations from the two n-best lists matches as specified above. Typical n values used in the subsequent sections are 1, 5, 10, 20, and 50.

ERG/ACE parseable metric is defined as the percentage of the HSSR realizations that can be parsed by the ERG/ACE parser. The metric measures the grammaticality of the HSSR output, since the ERG/ACE parser will fail to parse ungrammatical realizations. As with the ERG/ACE match metric, the ERG/ACE parseable metric can be generalized to n-best realizations of the HSSR system, where at least one of the realizations in the n-best list must be parseable.

7.2 Realization of gold DMRS representations

In the chapter introduction I described three realization scenarios considered in this thesis. In this section, I evaluate and compare the HSSR system to the established approaches on the first of the three scenarios, in which the DMRS representation is considered to be correct. In order to be considered correct, a DMRS representation needs to be validated via the process of treebanking by an expert (I will refer to DMRS representations obtained in this manner as *gold* representations). This scenario is well-suited to the established approach to realization, which is guaranteed to produce meaning-preserving grammatical outputs (in fact, it could be considered as the perfect scenario for the ERG/ACE realization systems). The approach presented in this thesis does not hope to surpass the output quality of the established approach on this task, but aims to come as close as possible. The objective of the experiment presented in this section is therefore to establish the relative performance of the realization systems introduced in Section 7.1.1 in this realization scenario.

The gold evaluation set used in this section consists of 90 treebanked (D)MRS representations.[6] Prior to treebanking, the English sentences were randomly sampled from the newstest2013 dataset.

The BLEU score and realization percentage results of the four systems on the gold evaluation set are reported in Table 7.1. The HSSR system achieves a high overall BLEU score

[6]The same gold dataset as used in Section 6.2.1 is used here. Eight sentences out of the total of 100 could not be treebanked under the ERG 1214, while the resulting DMRS graphs of two sentences exceeded the 20 node limit.

System	BLEU (brevity penalty)	BLEU on realized[8]	% realized (#)
HSSR	63.3 (1.0000)		100 (90/90)
ERG/ACE	65.7 (0.8812)		88 (80/90)
ERG/ACE++	68.8 (0.9334)	73.7 (1.0000)	92 (83/90)
ERG/ACE++HSSR	71.3 (1.0000)		100 (90/90)

Table 7.1: Gold DMRS realization result summary. Performance of the systems is characterized in terms of BLEU score and percentage of graphs realized.

and is able to realize all representations. As expected, the HSSR system is outperformed by the ERG/ACE system in terms of BLEU score (+2.4 BLEU points). However, the ERG/ACE system is unable to realize 12% of the representations. Its BLEU score is severely penalized because of it, as can be seen from the associated brevity penalty (see Section 6.2.2). The improved ERG/ACE++ system increases the number of sentences realized, which results in a 3.1 point BLEU score increase.[7] Despite the unknown word improvements of the ERG/ACE++ system compared to the ERG/ACE system, 8% of the representations could still not be realized. Finally, the combined ERG/ACE++ and HSSR system achieves the highest BLEU score (+2.5 BLEU points compared to ERG/ACE++ system) and realizes 100% of representations.

The ERG/ACE++ system failed to realize 7 out of 90 representations. In most cases, the reason for not realizing gold representations are missing trigger rules, which are responsible for realization of semantically empty words (discussed in Section 3.3.1).[9] On the other hand, the HSSR system is able to realize all 90 representations, demonstrating its robustness in this realization scenario. The best HSSR realizations of the seven representations that the ERG/ACE++ system was unable to realize are shown in Figure 7.3.[10] In comparison to the original treebanked sentence, examples 2 and 5 are perfect realizations, examples 1, 3, 6, and 7 are somewhat broken, whereas the realization in example 4 is severely broken.

An alternative measure of the HSSR system realization quality is the match percentage metric (introduced in Section 7.1.2), which measures the number of HSSR system realizations that (fuzzily) match the realizations of the ERG/ACE++ system. The match percentage results on the gold evaluation set are shown in Figure 7.4a. Around 23% of the 1-best HSSR realizations match the 1-best ERG/ACE++ realization. This is a surprisingly high number considering there are many ways of realizing a single representation. The 20 node limit may contribute to this somewhat, since shorter sentences will tend to have fewer possible realizations. We can also observe that the HSSR 1-best realization occurs in the ERG/ACE++ 5-best realizations around 34% of the time. However, increasing the size of the ERG/ACE++ n-best realizations beyond 5-best does not yield further improvements to the matched percentage. The size of the HSSR n-best list can

[7]Such dramatic BLEU point improvements are observed due to the small size of the dataset.

[8]'BLEU on realized' column refers to the BLEU score on the examples the system was able to realize, without being penalised by the BLEU score brevity penalty (see Section 6.2.2). This score is reported in relevant result tables for the ERG/ACE++ system.

[9]There is a long tail of infrequent trigger rules that are yet to be added to the ERG.

[10]Note that the HSSR realizations have only partial punctuation (see Section 3.3.1) and are in lower case (see Section 7.1.1), with the exception of the first character in a sentence, which is automatically capitalized during post-processing.

1. *And ourselves, with our 88 soldiers killed, plus the wounded, the maimed.*
   And we, with our 88 soldiers killed, plus the wounded, the maimed as

2. *Every year, 13 million migrant workers come to Moscow, St. Petersburg and other cities in Russia.*
   Every year, 13 million migrant workers come to moscow, st. petersburg and other cities in russia.

3. *But with time - as promised by the Federal Migration Service - tests will become mandatory for all non-residents.*
   But with time, as promised by the federal migration service, which tests will become mandatory for all non residents

4. *This is about 1.1% more than it spends on salaries right now.*
   This is about more than right now than it spends on salaries' is 1.1%

5. *Condoms can reduce the risk of contraction, however, they do not offer 100% protection.*
   Condoms can reduce the risk of contraction, however, they don't offer 100% protection

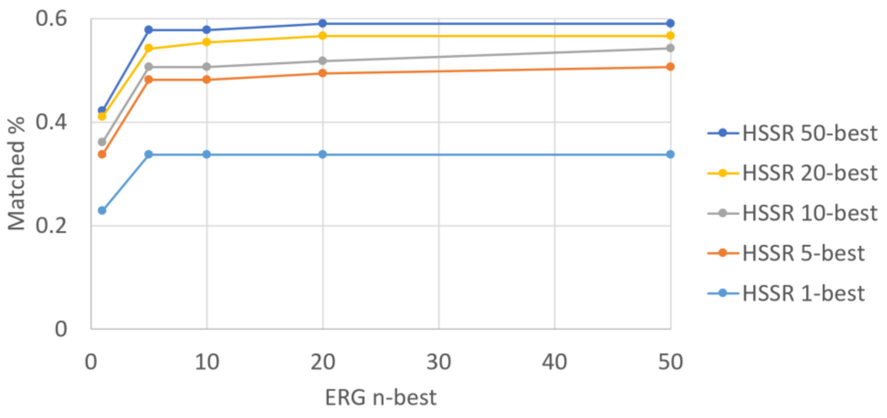
6. *It's all right for us - and our children - to be bored on occasion, they say.*
   They say that it is all right for us and our children bored on occasion

7. *It spends USD 1,300 per year on policies to encourage people to have more children.*
   $1,300 per year it spends on policies that encourage people to have more children.

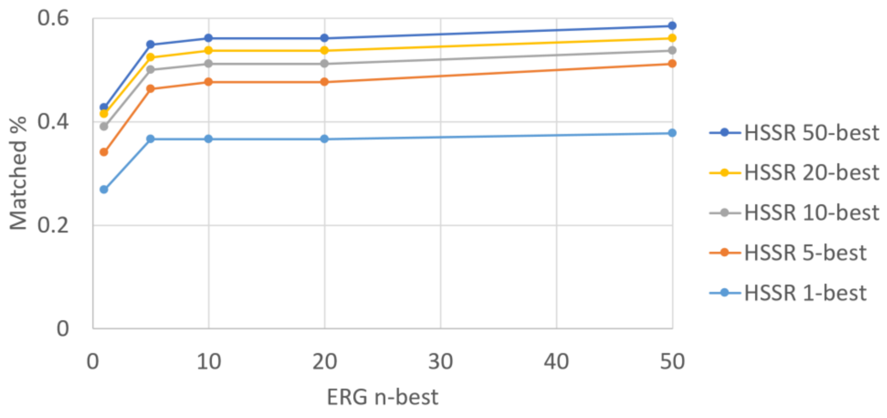
Figure 7.3: 1-best realization outputs of the HSSR system. The original treebanked sentences are shown in italics for reference.

also be increased, evaluating whether the HSSR system is capable of generating realizations matching the ERG/ACE++ system, but did not score them as highly. We can see that increasing the size of the HSSR n-best list gives significant but diminishing returns in terms of matching the realization outputs of the ERG/ACE++ system. In particular, matching the 5-best HSSR outputs against the ERG/ACE++ outputs yields a substantial improvement in the matched percentage (between 10-15 percentage points). The highest matching percentage, 59%, is achieved comparing the 50-best to a 50-best realization list.

In Figure 7.5, I compare a sample of the HSSR and ERG/ACE++ realizations to the original treebanked sentences. The first example shows a pair of identical realizations between the two systems (apart from an additional comma). The second example shows an example of the two systems generating different but correct realizations, neither of which are identical to the reference sentence. In fact, the 20-best list of the HSSR system contains both the reference sentence and the ERG/ACE++ realization. The 50-best ERG/ACE++ realization list contains the reference sentence, but not the HSSR realization. In the third example, the HSSR realization is substantially broken, while the ERG/ACE++ realization is somewhat marked but essentially correct in comparison with the reference sentence. Investigating the reason for the broken HSSR realization revealed that the HSSR system had at its disposal the carg rules for realizing 'I/43' and 'I/34' (see Section 6.1.3), but instead chose to use deletion rules (see Section 6.1.2). Finally, the fourth example shows



(a) Performance of the HSSR system in matching outputs of the ERG/ACE++ system on the gold dataset of 90 sentences.



(b) Performance of the HSSR system in matching outputs of the ERG/ACE++ system on the gold subset of the parsed newstest2013 dataset, corresponding to the 90 treebanked representations.

Figure 7.4: Performance of the HSSR system in matching outputs of the ERG/ACE++ system for two datasets. n-best outputs of both systems are considered for $n = \{1, 5, 10, 20, 50\}$..

an instance in which the ERG/ACE++ realization is arguably worse than the one by the HSSR system (largely, due to the inserted commas and better fluency of the HSSR system output).[11]

Overall, the HSSR realizations of gold representations are generally of high quality, but often noticeably worse than their ERG/ACE++ counterparts. However, despite the inputs being gold representations, the ERG/ACE++ system fails to realize 9% of them. Therein lies the strength of the HSSR system, since it is able to realize all gold representations. The combined ERG/ACE++HSSR system shares the strengths of both systems, indicating that in the case of gold representations, the HSSR system should be used in absence of a ERG/ACE++ realization.

[11]Admittedly, it was difficult to find an ERG/ACE++ realization of a gold representation that was worse than its HSSR counterpart.

1. *This summer's recruits are used to playing matches at a high level.*
   H/g This summer's recruits are used to playing matches at a high level.
   E/g This summer's recruits are used to playing matches, at a high level.

2. *They say, if you touch the walls on the street of your sign, fortune will come to you.*
   H/g Fortune will come to you if you touch the walls on the street of your sign, they say
   E/g They say that fortune will come to you if you touch the walls on the street of your sign.

3. *The I/43 and I/34 roads have been chemically treated around Svitavy.*
   H/g Chemical, the and roads have been treated around svitavy
   E/g The I/43 and I/34 roads have been treated around Svitavy, chemically.

4. *First: "army or military loot," i.e. weapons that were stolen during the fighting in the Caucasus.*
   H/g First: Army loot or military weapons that were stolen during the fighting in the caucasus.
   E/g First : army or military loot i. e. weapons, which, were stolen during the fighting in the Caucasus.

Figure 7.5: The 1-best realization outputs of the HSSR and ERG/ACE++ systems for four gold representations. The original treebanked sentences are shown in italics for reference, followed by the HSSR and ERG/ACE++ realizations.

7.3 Realization of ERG-parsed DMRS representations

In the previous section, I evaluated the four realization systems introduced in Section 7.1.1 in the first realization scenario (see Figure 7.2a), in which the representations are assumed to be correct. This scenario is not often encountered in practice, since it is rare for meaning representations to be treebanked by a human. Instead, the representations are commonly obtained via the ERG/ACE parser (briefly introduced in Section 3.1.3). Unlike the gold representations, representations obtained via parsing are subject to parsing errors, such as incorrect prepositional phrase attachment (PP attachment). ERG parsing is about as accurate as the best competing parsers (Ivanova et al., 2013). Parsing errors generally have adverse effects on ERG/ACE realization performance, since the incorrect interpretation of the original sentence will likely lead to incorrect realizations. On the other hand, the HSSR system is more robust to parsing errors, since it is trained on parsed examples and therefore has the capacity to 'overlook' common parsing errors. Namely, if a parsing error happens frequently, the HSSR system will have rules at its disposal to realize the intended meaning of the original sentence.[12]

[12]In practice, these rules are distinguished from the rules taking the meaning representation at face value only by the rule features (see Section 4.3.2) and it is therefore up to the log-linear model (see

System	BLEU (brevity penalty)	BLEU on realized	% realized (#)
HSSR	63.4 (1.0000)		100 (90/90)
ERG/ACE	63.4 (0.8569)		86 (78/90)
ERG/ACE++	67.3 (0.9206)	73.1 (1.0000)	91 (82/90)
ERG/ACE++HSSR	72.4 (1.0000)		100 (90/90)

Table 7.2: Realization result summary of the gold subset of the parsed newstest2013 dataset. Performance of the systems is characterized in terms of BLEU score and percentage of graphs realized.

In this section, I evaluate the performance of the realization systems on ERG-parsed representations. In the first experiment, presented in Section 7.3.1, I repeat the experiment from Section 7.2, only this time the 90 representations are derived from a parser instead of treebanking. Since the set of 90 representations is relatively small, I additionally evaluate the realization systems on two larger datasets (newstest2013 and newstest2014 initially introduced in Section 6.2.1) in Section 7.3.2. This evaluation gives a more reliable indication of relative realization system performance on parsed representations and is the realization equivalent to the translation system evaluation presented in Section 6.3.

7.3.1 Realization of ERG-parsed gold subset

In Section 7.2, I compared the four realization systems (introduced in Section 7.1.1) by realizing *gold* representations in a scenario where the representations are assumed to be correct. To achieve this, the 90 representations were treebanked by an expert human. In this section, I repeat the experiment from Section 7.2 on the 90 corresponding representations obtained via the ERG/ACE parser (I will refer to these as simply *parsed* representations).[13]

The results are shown in Table 7.2. The HSSR system achieves roughly the same performance as when realizing the gold representations (+0.1 BLEU score). Since the system was trained on parsed representations, this result is expected. As in the gold representation experiment, the system was able to realize all input representations. On the other hand, the performance of the ERG/ACE system decreased significantly (-2.3 BLEU points) in part due to realizing 2 representations fewer. The ERG/ACE++ system shows a similar trend in decreased BLEU score (-1.1 BLEU points) and 1 fewer representation realized. On the other hand, the performance of the combined system increased by 1.3 BLEU points.

The matched percentage of the HSSR realizations to the ERG/ACE++ realizations is shown in Figure 7.4b. Compared to the same metric on the gold representations (shown in Figure 7.4a), the matched percentage is higher with the parsed representations, but not significantly so. Increasing either the HSSR or ERG/ACE++ n-best list size leads to diminishing improvements over the performance on the gold representations. In fact, the

Section 5.4.3) to decide which interpretation to score highly.

[13]Note that the newstest2013 dataset was used to tune the HSSR system. However, as demonstrated in evaluation in Section 7.3.2, there isn't a significant difference in the HSSR system performance on the tuned dataset compared to a previously unseen dataset.

parsed realizations achieve the same matched percentage (59%) as gold realizations when comparing the two 50-best lists.

Instead of comparing the realizations between the two systems, the same matched percentage metric can be used to compare the realization outputs of a single system between the gold and parsed representations. The 1-best ERG/ACE++ realizations of the gold and parsed representations match in 80% of the cases, whereas they match in only 65% of the cases for the HSSR system. This indicates that the HSSR system is less constrained by the input representation. However, since its BLEU score remained roughly the same, the 35% of top (parsed) realizations that are different are on average of comparable quality to the gold realizations. Surprisingly, in 14% of the cases, there is not a single matching realization between 50-best realizations generated by the HSSR system. In ERG/ACE++, this only happens in 6% of the cases. This reaffirms the observation that the HSSR system is less constrained by the input representation - not only in the top realization, but in the top 50 realizations.

Example realization outputs are shown in Figure 7.6. In Figure 7.6a I show examples of the 1-best HSSR realizations that are different between gold and parsed representations. The examples correspond to the ones shown in Figure 7.3, which the ERG/ACE++ system was unable to realize. Compared to the realization produced from the gold representation, the first example is improved by dropping the unnecessary 'which'. The second example, which I deemed as severely broken in Section 7.2, is substantially improved when realized from the parsed representation. The same can be observed in the third example, which can now be considered a perfect realization when compared to the original sentence.

In Figure 7.6b I contrast the realization outputs of the HSSR and the ERG/ACE++ systems and compare them to their respective realization outputs of gold representations. In the first example, both the HSSR and ERG/ACE++ realizations of the parsed representation are worse than their gold counterparts (in the HSSR realization, correctly assigning the subject of 'to mine' is an improvement). The second example shows an interesting difference in realizations between the gold and parsed representations of the HSSR system: the parsed representation is realized in passive instead of active voice. On the other hand, the ERG/ACE++ realization of the parsed representation remains the same as with the gold representation. In the final example, the HSSR realization of the parsed representation is better than its realization of the gold representation due to not dropping the word 'shudder'.

Based on the experimental results and the analysis in this section, we can conclude that the HSSR system maintained its performance in realizing parsed representations in comparison to realizing gold representations. On the other hand, the ERG/ACE and ERG/ACE++ system performance decreased, in part due to fewer realized representations. Comparison of the realizations of the same system between two sets of representations indicated that the HSSR system is more sensitive to the input representations and its realizations tend to vary more. However, this did not result in degraded performance, as indicated by the consistent BLEU score. Instead it results in alternative realizations of similar quality, which we observed in the example realizations.

7.3.2 Realization of ERG-parsed datasets

The previous two realization experiments described in Section 7.2 and Section 7.3.1 were conducted on a small dataset of 90 representations. Due to the small size of the dataset,

1. *But with time - as promised by the Federal Migration Service - tests will become mandatory for all non-residents.*
   H/g But with time, as promised by the federal migration service, which tests will become mandatory for all non residents
   H/p But with time, as promised by the federal migration service tests will become mandatory for all non residents

2. *This is about 1.1% more than it spends on salaries right now.*
   H/g This is about more than right now than it spends on salaries' is 1.1%
   H/p This is right now, about 1.1 percent more than it spends on salaries

3. *It spends USD 1,300 per year on policies to encourage people to have more children.*
   H/g $1,300 per year it spends on policies that encourage people to have more children.
   H/p It spends $1,300 per year on policies that encourage people to have more children.

(a) Comparison of the 1-best HSSR realizations of the gold (top) and parsed (bottom) representations.

1. *People come like hungry fish to bait, and then mine coal in Siberia.*
   H/g People come to bait like hungry fish and then coal is mined in siberia
   H/p People come to like to be baited, and then to mine coal in siberia hungry fish
   H/p People come to bait, like hungry fish, and then mine coal, in Siberia.
   E/p People come to like hungry fish to bait and then mine coal in Siberia.

2. *The Internet has caused a boom in these speculations.*
   H/g The internet has caused a boom in these speculations
   H/p A boom in these speculations have been caused by the internet.
   E/g The Internet has caused a boom in these speculations.
   E/p The internet has caused a boom in these speculations.

3. *The atmosphere would have made a Stalinist shudder with apprehension.*
   H/g The atmosphere, which would have made a stalinist with apprehension
   H/p It would be the atmosphere to have made a stalinist shudder with apprehension
   E/g The atmosphere would have made an Stalinist shudder with apprehension.
   E/p The atmosphere would have made a stalinist shudder, with apprehension.

(b) 1-best example realizations of the HSSR (*H*) and ERG/ACE++ (*E*) systems, compared to the reference sentence. For each of the two systems, the 1-best realization of the gold (*g*) and the parsed (*p*) representation is shown.

Figure 7.6: Examples of 1-best realization generated by the HSSR and ERG/ACE++ systems.

	newstest2013			newstest2014		
System	BLEU	BLEU on realized	% realized	BLEU	BLEU on realized	% realized
HSSR	63.6		100	63.4		100
ERG/ACE	51.0		76	49.5		75
ERG/ACE++	65.7	68.7	93	65.2	66.4	93
ERG/ACE++HSSR	67.6		100	66.0		100

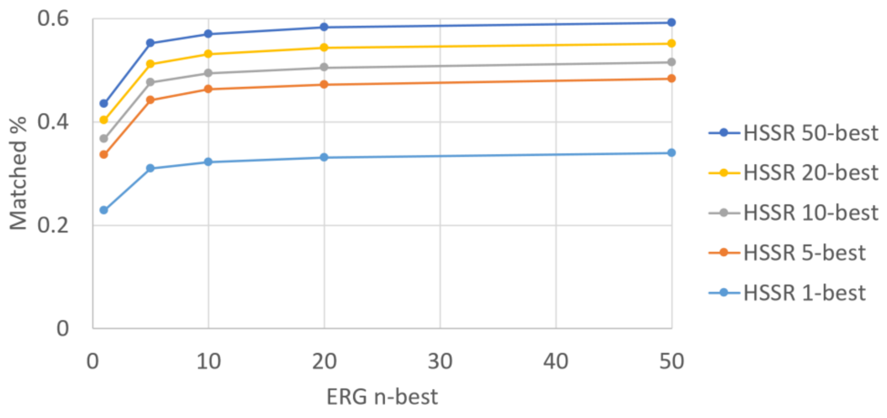
Table 7.3: Parsed DMRS realization result summary on the large newstest2013 and newstest2014 datasets. Performance of the systems is characterized in terms of BLEU score and percentage of graphs realized.

minor changes in realization quality and in the number of realized sentences resulted in large changes in BLEU score. In this section, I compare the four realization systems on two larger parsed datasets in order to obtain a more reliable relative performance comparison. The newstest2013 and newstest2014 datasets (consisting of 1774 and 1574 graphs) were initially introduced in Section 6.2.1 (later in the context of realization in Section 7.1.1) and previously used to evaluate the performance of the HSST system on the translation task (see Section 6.3).[14]

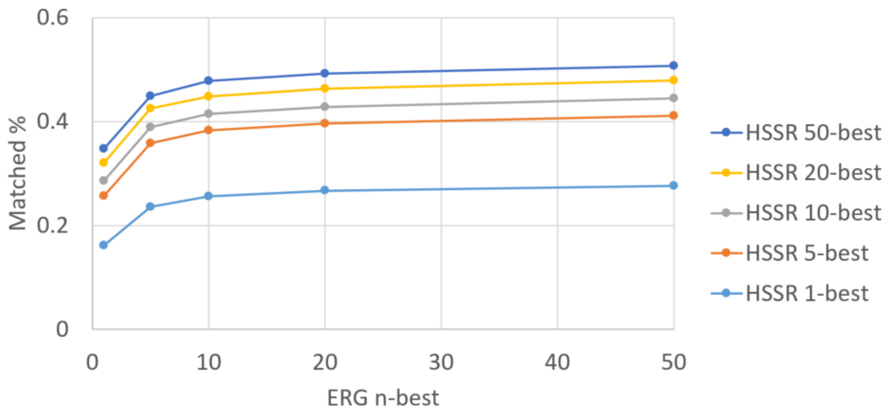
The realization performance on the two datasets is summarized in Table 7.3. The performance of the HSSR system in terms of BLEU score is comparable to the performance reported in experiments described in the previous sections. Additionally, the HSSR system continues to demonstrate its robustness by realizing all input representations. On the other hand, the realization percentage of the ERG/ACE system is significantly lower than in the previous experiments (76% and 75% on the current two datasets compared to 88% and 86% in the previous two experiments). Consequently, its BLEU score is significantly lower as well (-12.6 BLEU points compared to the HSSR system). However, the ERG/ACE++ system maintained a similar realization percentage as in the two previous experiments (93%), indicating that the ERG/ACE system suffered low realization percentage primarily due to unknown words occurring frequently in the two larger datasets. The ERG/ACE++ system BLEU score remains higher than the HSSR BLEU score, but the relative difference is smaller (+2.1 and +1.8 on newstest2013 and newstest2014 versus +5.5 and +3.9 points in the previous experiments). Finally, the combined system ERG/ACE++HSSR remains the strongest system with the highest reported BLEU scores and realizing all input representations of both datasets. Compared to the variations in BLEU score of the ERG/ACE systems, the HSSR system demonstrates very consistent performance. This may be due to the HSSR parameter tuning, which optimizes the BLEU metric, whereas the ERG/ACE systems do not.

The matched percentages between the HSSR and ERG/ACE++ systems for newstest2013 and newstest2014 datasets are shown in Figure 7.7. The matched percentages for the newstest2013 dataset are similar compared to the percentages reported for the subset of newstest2013 dataset. However, there is a significant difference between the matched percentages between the newstest2013 and newstest2014 datasets. This may be because newstest2013 was used for tuning or simply due to the differences in the datasets themselves. Due to the increased size of the datasets, the matched percentages for both datasets

[14]Additionally, the newstest2013 dataset was used to tune the HSSR system parameters.



(a) Performance of the HSSR system in matching outputs of the ERG/ACE++ system on the parsed newstest2013 dataset.



(b) Performance of the HSSR system in matching outputs of the ERG/ACE++ system on the parsed newstest2014 dataset.

Figure 7.7: Performance of the HSSR system in matching outputs of the ERG/ACE++ system for the two parsed datasets. n-best outputs of both systems are considered for $n = \{1, 5, 10, 20, 50\}$.

are smoother in comparison to the previous experiments (shown in Figure 7.4), especially with increasing size of the ERG/ACE++ n-best list.

There is a match for the top HSSR realization in the top 50 ERG/ACE++ realizations 34% of the time in the newstest2013 dataset. The corresponding number for the newstest2014 dataset is 28%. In order to characterize the quality of the remaining 66% and 72% HSSR realizations, I computed how many of the unmatched top HSSR realizations can be parsed with ERG/ACE (the ERG/ACE parseable metric described in Section 7.1.2). 77% of unmatched 1-best HSSR realizations for newstest2013 can be parsed, and 80% for newstest2014, indicating a high degree of grammatical outputs of the HSSR system (furthermore, there is a parseable realization in the top five HSSR unmatched realizations 89% of the time for both datasets[15]). This leaves 18% of the top HSSR realizations of both newstest2013 and newstest2014 which are neither matched by the ERG/ACE++ 50-best realizations nor parseable with ERG/ACE.

The results discussed above are in large part consistent with the previous two experiments,

[15]Hence, ERG parsing could be used as a filter to avoid very broken HSSR outputs.

but are more reliable because of the larger size of the two datasets used. The HSSR system remained consistent in the BLEU score achieved and in realizing all input representations. On the other hand, the ERG/ACE system performed significantly worse than previously due to unknown words. This deficiency is addressed in the improved ERG/ACE++ system, which outperforms the HSSR system in terms of BLEU score, but is still unable to realize 7% of input representations. As before, the best results on parsed data are achieved by the combined system.

7.4 Realization of non-ERG representations

In the previous two sections (Sections 7.2 and 7.3), the HSSR system proved to be capable of robustly producing high quality outputs. Unlike the established approach, it is able to realize all input representations, while the quality of its realizations comes close to the established approach on both gold and ERG-parsed representations. However, while ERG-parsed representations may be flawed due to parsing errors, both gold and ERG-parsed representations are well-formed. In this section, I consider representations which may not be only flawed but also not well-formed. Such representations are often a result of an external system, which either creates the representation or modifies an existing representation for a particular application. I refer to such representations as *non-ERG*. Realizing non-ERG representations constitutes the third realization scenario (shown in Figure 7.2c). As discussed and observed previously, the reliance on the grammar means that the ERG/ACE realizer is brittle when presented with non-ERG representations. Instead of producing realizations of degraded quality, the established approach fails to produce any output at all. The HSSR system, on the other hand, is more robust, as evident from the experiments in the previous sections, and therefore capable of realizing such inputs.

I demonstrate the robustness of the HSSR system on representations in three experiments. In Section 7.4.1, I consider a sentence simplification application, which removes significant parts (nodes and edges) of input DMRS graphs in order to simplify the original sentence. I show that the HSSR system is capable of realizing such heavily modified representations. Robust parsing with a PCFG is a recent addition to the DELPH-IN arsenal that enables parsing of sentences that were previously out of coverage. However, despite being useful for some applications, the resulting representations are often not well-formed. In Section 7.4.2, I demonstrate that the HSSR system achieves significantly better performance than the established approach on realizing representations resulting from robust parsing. Finally, in Section 7.4.3, I demonstrate that using the HSSR system as the realization component has the potential to improve performance and coverage of a transfer-based machine translation system.

7.4.1 Realization for sentence simplification

Sentence simplification is the task of shortening the sentence by removing non-essential information while preserving the main message of the sentence. A variant of the task is sentence compression, in which a sentence is simplified only by removing tokens so that the simplified sentence is a token subsequence of the input sentence. A recent successful approach to the sentence compression task by Filippova et al. (2015) uses LSTMs to delete tokens in the input sentence (refer to their related work section for a review of previous approaches). Sentence compression has been used to improve other tasks. For instance, Hasler et al. (2016) improve translation of long sentences with complex syntax and long-distance dependencies by using translations of a simplified sentence to guide the translation of the full sentence.[16]

Of particular interest is an approach to sentence compression attempted by Eva Hasler for the purpose of improving SMT (as in Hasler et al. (2016)), which makes use of DMRS representations.[17] Sentence compression is achieved by parsing the source sentence with ERG/ACE, using graph modelling methods described in Chapter 3, and then compressing it by removing nodes and edges using a set of heuristics. The compressed sentence is obtained from the tokens aligned to the remaining nodes (the heuristic token alignments are obtained as described in Section 3.3.1).

Instead of using the aligned tokens to obtain the simplified sentence, the compressed DMRS graphs can be realized with the HSSR system presented in this thesis. Such an approach can no longer be considered as sentence compression, since the realization system is not constrained to produce a token subsequence of the original sentence. Using a realization component instead allows for paraphrasing of the original sentence, which could be considered as a broader task of sentence simplification. However, the distinction is not important for the purposes of this thesis. Rather, our interest is in demonstrating the ability of the HSSR system to realize heavily modified DMRS graphs resulting from sentence compression.

In the remainder of this section, I briefly summarize the DMRS-based sentence compression approach. I continue by discussing the performance of the HSSR system in realizing compressed DMRS graphs and comparing the realized sentences to simplified sentences obtained via token alignment. I do not compare the HSSR system to the ERG/ACE systems since the compression heuristics rely on graphs modified with the methods described in Chapter 3. The changes to the graphs make them unrealizable by the ERG/ACE system.

The DMRS graph compression approach introduced above consists of a series of steps which remove nodes and edges from the original graph. The steps include: (1) removal of constructions such as ‘said X’, ‘explained X’ etc.; (2) removal of the right sides of coordinated constructions; (3) removal of relative clauses; (4) removal of appositions; (5) removal of nodes modifying the main verb of the sentence; and (6) removal of nodes connected via `?/EQ` edge (under certain conditions). Finally, all nodes that are disconnected from the main part of the graph as a result of these steps are also removed. An example of a sentence compressed with the DMRS-based approach is shown in Figure 7.8.

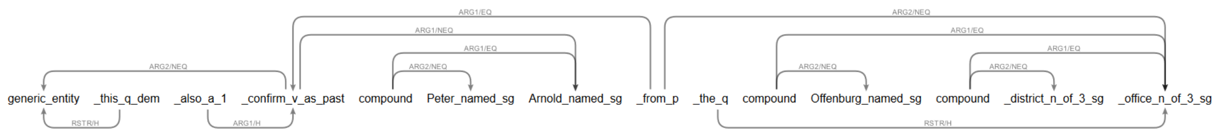
Using the above approach I compressed 1574 graphs of the newstest2014 dataset. I subsequently realized the graphs using the (unmodified) HSSR realization system presented in Section 7.1.1. For comparison, I created compressed sentences from heuristically aligned tokens, as in the original approach using the heuristically aligned tokens (I refer to it as *align* system). Both approaches reduced the length of the original sentences by around a half. The compression ratio, defined as the number of tokens after simplification divided by the number of tokens before simplification computed over the entire dataset, is 50.8%

[16]Note that in their work, Hasler et al. (2016) use manually created simplified sentences instead of creating them with an automatic approach, which they leave for future work.

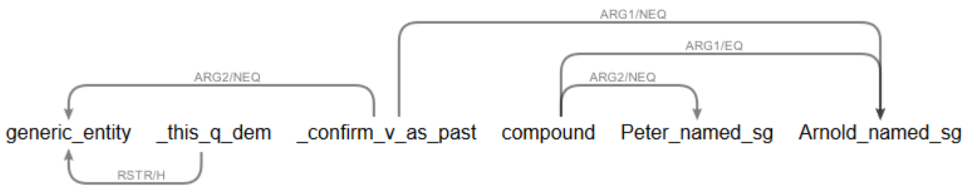
[17]Unpublished work at the time of writing.

This was also confirmed by Peter Arnold from the Offenburg District Office.

(a) The uncompressed sentence.



(b) The DMRS graph of uncompressed sentence after DMRS graph modelling steps.



(c) The compressed graph.

This was confirmed by Peter Arnold.

(d) The compressed sentence obtained from (c).

Figure 7.8: An example of sentence compression using the DMRS-based approach and token alignment.

for the HSSR system and 52.4% for the align system. The similar compression ratios are expected since the systems share the DMRS-based compression algorithm. The HSSR system likely has a slightly lower compression ratio due to node deletion in cases where it could not realize parts of the compressed DMRS graph.

The two systems often produce identical simplified sentences: 41.2% of sentences match when compared using the matched percentage metric (introduced in Section 7.1.2). I highlight some differences between the approaches by examining example outputs shown in Figure 7.9. Additional example outputs are shown in Appendix A.4.

The HSSR is capable of reordering or paraphrasing the sentence (example 1 and 2). However, occasionally, the reordering yields a worse results (example 3). The token alignment is a heuristic approach, which means that it may omit tokens. This can be observed in example 4 in which the 'of' token is omitted from the compressed sentence. The HSSR system is also able to create a fluent sentence from a heavily compressed DMRS (example 5 and 6).

The HSSR system makes some frequent mistakes. When the grammar does not contain a rule to realize a node, it instead uses its deletion rules (example 7). This contributes to the robustness of the HSSR system, but in the case of simplification results in a more compressed output then intended by the DMRS-based algorithm. Sometimes, the deletion is not as elegant as in the above example and instead results in a broken sentence (example 8). Another type of mistake observed several times is failing to realize negation, which severely affects the meaning of the sentence (example 9). The HSSR system also occasionally inserts additional words when they are not needed (example 10).

In conclusion, the HSSR system demonstrated its robustness by realizing heavily modified DMRS graphs for the sentence simplification task. In comparison to the alignment approach, the HSSR system enables paraphrasing and reordering, which is a departure

1. *They were unable to visit two sites because of security concerns, the inspectors said.*
   - *H* The inspectors said they were unable to visit sites.
   - *A* They were unable to visit sites the inspectors said.

2. *Frankfurt parking fees to increase dramatically*
   - *H* Frankfurt parking fees to increase dramatically
   - *A* Dramatically increased frankfurt parking fees

3. *The government in Jerusalem fails to confirm an attack on the Syrian airforce base*
   - *H* The government fails to confirm an attack
   - *A* Fails to confirm an attack the government.

4. *According to accusations, this was part of a turf war and represented a demonstration of power.*
   - *H* This was part of a turf war.
   - *A* This was part a turf war.

5. *Bamford is appealing the sentence and has been granted bail of 50,000 baht.*
   - *H* The sentence is being appealed.
   - *A* Is appealing the sentence.

6. *It makes me yearn for the many promises that disappeared.*
   - *H* I yearn for the promises
   - *A* Me yearn for the promises.

7. *"New Express's editorial management was disordered," the regulator said in a statement.*
   - *H* New express's editorial management.
   - *A* New express's editorial management was disordered.

8. *While Murphy said the purpose of the trip is to help improve relationships, he said some "tough love" will also be dispensed.*
   - *H* He said that also, some tough love.
   - *A* He said some tough love will also be dispensed.

9. *It was not something people wanted.*
   - *H* It was something
   - *A* It was not something.

10. *Aircraft electronic device rules to stay in force in Australia for now*
    - *H* Aircraft electronic device rules, for now, though, to stay in force in australia
    - *A* Aircraft electronic device rules to stay in force in australia for now

Figure 7.9: Sentence simplification examples from the newstest2014 dataset. The original sentence is shown in italics and two simplified versions of it are shown below (top - simplified with the HSSR system; bottom - simplified with the align system).

from the strict sentence compression task. It is important to note that the HSSR system was not adjusted for the sentence simplification task. For instance, in order to make it more suitable for the simplification task, the system parameters could be tuned on gold simplified sentences. The entire machine translation approach introduced in this thesis could potentially be used to train a simplification system. Additionally, a modification of the existing rule constraints (see Section 4.2.3) would allow learning of explicit deletion rules, as opposed to using only general deletion rules (introduced in Section 6.1.2). A similar approach to sentence simplification using statistical machine translation (specifically, tree-to-tree transducers) was described by Cohn and Lapata (2008).

7.4.2 Realization for robust parsing

One of the downsides of ERG/ACE parsing (briefly described in Section 3.1.3) is its coverage. The parser is unable to parse a proportion of input sentences either (1) due to their ungrammaticality and/or unusualness, or (2) due to errors in the grammar. The parsing coverage can be increased by allowing informal roots, which I used in parsing for the experiments presented in this thesis (see Section 6.2.1). The resulting parsing coverage is around 81% for training data and 94% and 93% for newstest2013 and newstest2014 respectively (see Section 6.2.1 for full details).

Recently, robust ERG/ACE parsing using a probabilistic context-free grammar (PCFG) was introduced by Packard and Flickinger (2017). With robust parsing, an analysis can be produced even when the input sentence is not well-formed. Using the PCFG approach, 100% parsing coverage can be achieved. The downside of robust parsing with a PCFG is that, although the output is a useful MRS, it may not be completely well-formed. As discussed and demonstrated in this chapter, the established ERG/ACE realization approach is in large part incapable of realizing MRS representations that are not well-formed. Consequently, the robust parsing approach is limited in regeneration scenarios, since the resulting MRS representations cannot be used to produce target sentences (e.g., after modification). The HSSR system, on the other hand, is not so sensitive to the well-formedness of the meaning representations and is therefore capable of realizing the representations created by the robust parser. In the remainder of this section I briefly describe robust PCFG parsing, followed by a description of the experimental setup and a discussion and comparison of realization system performance on the representations obtained with the robust parsed using a PCFG.

Summarizing the presentation by Packard (2016b), robust parsing uses a probabilistic context-free grammar as an alternative to parsing sentences with an HPSG. Following the work of Zhang and Krieger (2011), the PCFG is estimated on the derivation trees obtained from the existing (HPSG-based) ERG/ACE parser, in addition to the gold treebanks. Parsing a sentence with a PCFG results in a derivation tree which usually fails to unify. Therefore, the robust parsing approach has a second part, robust unification, which enables a feature structure to be constructed. The resulting feature structure may not be well-formed, but contains a useful MRS representation. By adjusting the PCFG estimation, it is possible to achieve 100% parsing coverage. There are two downsides of robust parsing: (1) the parsing accuracy of parses obtained in a robust way is lower compared to the standard ERG/ACE parser (there is a trade-off between parsing accuracy and coverage), and (2) using a large PCFG for robust parsing results in a dramatic

	newstest2013r			newstest2014r		
System	BLEU	BLEU on realized	% realized	BLEU	BLEU on realized	% realized
HSSR	45.3		100	50.7		100
ERG/ACE	0.1		11	2.8		26
ERG/ACE++	0.2	55.3	12	3.4	51.2	28
ERG/ACE++HSSR	46.1		100	50.2		100

Table 7.4: Result summary of realization for robust parsing. Performance of the systems is characterized in terms of BLEU score and percentage of graphs realized.

increase in parsing time. The latter issue can be partially addressed via pruning with supertagging.

In this section, I evaluate the realization performance of the four realization systems (presented in Section 7.1.1) on the representations obtained with the robust parser. In particular, I robustly parse the sentences of the newstest2013 and newstest2014 datasets that could not be parsed with the regular ERG/ACE parser. The PCFG[18] used for robust parsing in this section was trained on 100 thousand treebanked sentences (gold) and 50 million ERG/ACE parsed sentences of the WikiWoods corpus (Flickinger et al., 2010). Apart from specifying the additional PCFG setting, the same ERG/ACE parsing settings as listed for newstest2013 and newstest2013 datasets in Section 6.2.1 were used. The same graph modelling steps as described previously were used to process the DMRS graphs after the MRS to DMRS conversion. In line with previous experiments, the final DMRS graph set was filtered to contain graphs with at most 20 nodes, yielding 71 and 75 graphs for newstest2013 and newstest2014 datasets respectively. I will refer to the new datasets as newstest2013r and newstest2014r to avoid confusion.

The evaluation results are summarized in Table 7.4. The HSSR system significantly outperforms the ERG/ACE and ERG/ACE++ systems on both datasets. The two ERG/ACE systems are outperformed primarily due to their inability to realize representations that are not well-formed. In total, they are unable to realize between 70 and 90 percent of the representations, whereas the HSSR system is capable of realizing all of them. There is only a minor improvement in the performance of the ERG/ACE++ system compared to the ERG/ACE system. However, there is a significant difference in terms of realization quality between the two datasets for all realization systems, which can only be explained by the differences in the datasets themselves (for instance, the ERG/ACE system is able to realize 26% of the newstest2014r and only 11% of the newstest2013r). We can also observe that the combined system is not necessarily the best option: while it is the best performing system on newstest2013r, it is worse compared to the HSSR system on the newstest2014r dataset (although the difference is only 0.5 BLEU point). Example realizations are shown in Appendix A.5.

7.4.3 Realization for transfer-based MT

Transfer-based machine translation is an important application of MRS and related technologies (part of the DELPH-IN consortium) with a long history (Copestake et al., 1995;

[18]Originally downloaded from http://sweaglesw.org/linguistics/csaw/download/ww-1214-gp2.pcfg.bz2

Oepen et al., 2004a; Bond et al., 2005; Oepen et al., 2007; Nichols et al., 2007; Bond et al., 2011). I reviewed most notable approaches to translation with MRS in Section 2.4. The common goal of transfer-based translation is to produce high quality outputs. This comes at the expense of translation coverage. For example, the Japanese-English transfer-based MT system presented by Bond et al. (2011) produces translations for 26% of input Japanese sentences. In order to alleviate the coverage problem, the transfer MT system can be combined with a back-off SMT system (Nichols et al., 2007; Bond et al., 2011). However, this is not an ideal solution, since the two systems are completely decoupled and the SMT system does not benefit from any part of the transfer system.

Each stage of the parse-transfer-realize pipeline is responsible for a reduction in the final end-to-end coverage. In particular, Bond et al. (2011) report a 54% realization coverage (in comparison, parsing and transfer achieve 80% and 60% coverage respectively). The low realization coverage is largely a consequence of malformed MRS representations, created via semantic transfer. In order to improve the transfer-based MT system coverage and consequently performance, the robust HSSR system can be used instead of or in combination with the high precision but brittle ERG/ACE realizer. With 100% realization coverage (extrapolating the HSSR performance observed in previous experiments presented in this chapter), the transfer MT system presented in Bond et al. (2011) would achieve end-to-end coverage of 48%, an increase of 84%.

In order to evaluate the capabilities of the HSSR system for realization of transferred representations, I setup a simplified Japanese-English transfer pipeline (in comparison with the system presented in Bond et al. (2011)) and compare the realization system performance on the transferred English MRS representations in terms of BLEU score and coverage achieved.

The transfer-based MT system used in the experiments in this section uses the JACY grammar[19] (Siegel et al., 2016) in YY mode, using Japanese segmentation and tokenization.[20] Japanese-English transfer grammar[21] consists of hand-crafted transfer rules and a large number of transfer rules automatically obtained from bilingual dictionaries and parallel corpora (see Section 2.4). I introduce several simplifications in comparison to the transfer system of Bond et al. (2011), which make the experiments easier to set up, more tractable, and fairer for realization component comparison. Instead of considering 5-best outputs for each input at every step, I only consider 1-best output after every step to improve ease of setup and tractability. Consequently, each realization system is presented with a single transferred English MRS representation for each input Japanese sentence and only the 1-best realization output is considered. Since there is only one realization output, I also do not use the end-to-end reranking component. The omission of end-to-end reranking is also due to fairness of realization system comparison, since the discriminative model was trained with the ERG realization system.[22] Finally, I do not use a back-off SMT system. These simplifications result in a Japanese-English transfer system that is significantly weaker than reported in Bond et al. (2011). Regardless, this should not meaningfully affect the primary purpose of the experiments reported in this

[19]JACY grammar was obtained from https://github.com/delph-in/jacy. Latest change to the grammar was made on November 30 2016 (commit hash 41c6ea4).

[20]MeCab version 0.996, available at http://taku910.github.io/mecab/.

[21]Japanese-English transfer grammar is available at http://svn.emmtee.net/trunk/uio/tm/jaen/ (revision 25328).

[22]An HSSR-friendly end-to-end reranking model could be trained with the HSSR realization system outputs, but I consider that to be out of the scope of this thesis.

section, which is the evaluation and comparison of the realization systems on transferred MRS representation inputs.

I realize transferred representations with the four realization systems described in Section 7.1.1. Unlike in previous experiments presented in this chapter, the systems need to be augmented in order to make them suitable for realizing transferred representations:

- Instead of using ERG version 1214, both the ERG/ACE and the ERG/ACE++ systems use ERG version 1212 in experiments reported in this section. This is because the Japanese-English transfer grammar has not been updated with the latest ERG 1214 changes, meaning that both systems achieve better coverage with the older grammar. The same issue affects the HSSR system but I opted not to address it.[23]
- When the transfer component does not have a lexical rule for transferring a Japanese predicate, the predicate is transferred in its Japanese form (e.g., `ja:_oyogi_n_1_rel`). Neither the ERG/ACE nor the HSSR system are able to realize such predicates (the ERG/ACE realization fails whereas the HSSR system is forced to delete the predicate). I introduced preprocessing steps for both the ERG/ACE++ and the HSSR system that allow them to pass the Japanese predicate lemma to the target side (e.g., translated as 'oyogi').[24]
- The 'ja:' prefix of transferred grammar predicates is removed for both realization systems and some common transferred grammar predicates, which cause the ERG/ACE realization failure and/or HSSR deletion, are mapped or deleted from the representation.
- The HSSR system is unable to realize nodes with underspecified properties (e.g., Japanese third person pronoun without a specified number) since it has never observed underspecified labels in the training data. As an approximate solution, underspecified node properties are specified as the most frequent value for the given property (e.g., underspecified number is set to singular).[25] In order to enable realization of underspecified variables in the ERG/ACE systems, I disabled the ACE subsumption test.
- Since the ERG/ACE system is sensitive to the quality of the transferred MRS, I use it as a pre-selection criterion for the other realization systems. Namely, if a particular representation can be realized with ERG/ACE, that representation is subsequently realized by the ERG/ACE++ and the HSSR systems. Similarly, if none of the transferred representations could be realized by the ERG/ACE system but

[23]The HSSR system was trained on representations obtained with ERG 1214. In order to address the ERG version mismatch and obtain the best realization performance, an HSSR grammar could be extracted from training examples obtained with ERG 1212. However, since the HSSR system is more robust to diverging inputs, I opted not to extract a new grammar.

[24]The ERG/ACE++ system uses a similar approach as when handling unknown words, described in Section 7.1.1; the HSSR system treats the lemma as a carg and consequently allows it to be realized with a carg rule (see Section 6.1.3).

[25]A more principled approach would consider translations of all possible specified values. A similar problem is caused by overly specified properties (e.g., Japanese second person singular pronoun, with masculine or feminine gender), although a solution - removing the irrelevant information - is straightforward.

	Coverage %				BLEU	BLEU on realized
Realization system	Parse	Transfer	Realization	End-to-end		
HSSR			98.6	70.7	3.7	
ERG/ACE	80.7	88.9	10.7	7.7	0.0	
ERG/ACE++			29.4	21.1	0.6	7.2
ERG/ACE++HSSR			98.6	70.7	4.5	

Table 7.5: Japanese-English translation coverages of a transfer-based system using different realization systems.

a representation could be realized by the ERG/ACE++ system, it is pre-selected to be realized by the HSSR system.

The four realization systems are compared on the first 1000 sentences of the development part of the Japanese-English Tanaka corpus (Tanaka, 2001) used in Bond et al. (2011). Since the sentences in the Tanaka corpus tend to be short, they were not filtered based on their respective node set size. Instead, the HSSR system filtered the DMRS graphs with more than 20 nodes internally.

The coverage results are shown in Table 7.5. We can observe that the transfer coverage is considerably higher than the 60% coverage reported by Bond et al. (2011), while the ERG/ACE and ERG/ACE++ realization coverages are both significantly lower. This is likely a consequence of several factors. The most important factor is likely to be the missing well-formedness check after the transfer component of the pipeline, which would reject ill-formed transferred MRS before attempting to realize them.[26] Another contributing factor is the changes to the JACY and the ERG grammars since the last update to the Japanese-English transfer grammar, which yield small incompatibilities between the transfer pipeline components. Compared to the ERG/ACE and ERG/ACE++ systems, the HSSR system achieves a significantly higher realization coverage, which results in a significantly higher end-to-end coverage. I do not compare the realization system BLEU scores due to low coverages of the ERG/ACE and ERG/ACE++ systems. Instead, I examine the quality of the outputs qualitatively below.

In Figure 7.10 I contrast example HSSR realizations with corresponding ERG/ACE realizations and compare them to the reference translations. The HSSR system performs well in many instances compared to the ERG/ACE system (examples 1, 2, and 3) by producing more fluent and grammatical realizations closer in meaning to the reference sentence. The transferred representations preserve the meaning of the original sentence to a varying degree. When they do not preserve it well, neither of the systems is capable of producing realizations similar to the reference sentences (examples 4, 5, and 6). A weakness of the HSRR system is incorrect choice of possessive pronoun and subject-verb agreement, which the ERG/ACE system has comparatively few problems with (examples 7 and 8). The HSSR system uses its deletion rules in order to realize predicates for which it has no rules in the realization grammar, which results in realizations with missing words (examples 9 and 10). The HSSR realization in example 9 also exhibits the heuristic of specifying the

[26]In Bond et al. (2011), ERG realization is used as a filter of ill-formed transferred MRS (personal communication with Francis Bond).

underspecified determiner to the most frequent determiner (*the*), which is an incorrect choice in this instance.

Even though in some of the above examples the transferred representations did not preserve the meaning of the original sentence, they were sufficiently well-formed to be realized by the ERG/ACE system. This only happened in 10.7% of the cases, as shown in Table 7.5. In further 18.7% of the cases, the transferred MRS representations were realized with the improved ERG/ACE++ system, but not with the ERG/ACE system. Examples of such HSSR and ERG/ACE++ realizations are shown in Figure 7.11a. Both systems are capable of producing realizations of relatively high quality (example 1 and 2). Many of these representations contain untransferred Japanese predicates, which the ERG/ACE++ system (like the HSSR system) realizes directly (examples 3 and 4). The HSSR system is prone to occasionally producing an unusual realization (example 5).

Finally, in Figure 7.11b I show example HSSR realizations for which no corresponding ERG/ACE or ERG/ACE++ realizations were produced. The HSSR system can produce high quality realizations from ill-formed inputs (example 1), but more commonly it produces realizations of degraded quality (examples 2 and 3). As in examples in Figure 7.11a, the realizations are interspersed with Japanese lemmas due to untransferred Japanese predicates (examples 4 and 5). When the system cannot realize a predicate node, it deletes, which results in shortened output (example 6) as observed previously. Occasionally, it also produces somewhat humorous realizations (example 7). Additional example realizations are shown in Appendix A.5.

In conclusion, the transfer MT experiment reported in this section demonstrated the ability of the HSSR system to realize representations of varying degrees of well-formedness into realizations of potentially degraded quality. Furthermore, the coverages reported in Table 7.5 and the evidence observed from analysing the realization outputs indicate that the HSSR system has good potential to be used as the realization component of a transfer-based translation system. However, more extensive experiments with a non-simplified transfer pipeline are required in order to make a conclusive claim.[27] Notably, the HSSR system has not been adapted to transfer MT realization task in a significant way (apart from the heuristic adjustments discussed above). Further realization performance improvements are likely to be obtained by extending the SCFG with rules extracted for the specific transfer MT realization task. In particular, rules extracted from (transferred MRS, reference translation) pairs would allow the HSSR system to learn to accommodate the systematic errors made by the transfer component.

[27] I consider such experiments to be out of the scope of this thesis.

1. *He waited on his master.*
H Served as him, to that master.
E He served to that master.

2. *I walked along the street.*
H I walked along the street
E I walked me, along the street.

3. *She was displeased at my letter.*
H She saw the letter of me and it took offense
E I saw a letter of me, and they took her offense.

4. *The plague was about that year.*
H That year was caught by the plague.
E A plague caught that year.

5. *She was injured in the car accident.*
H As that auto accident, she injured
E Injured her, as that automobile accident

6. *Do your best!*
H Yourself, do the time.
E Do the goods.

7. *His manner to us was kind.*
H He's attitude, which is gentle
E His attitude was gentle.

8. *Tadpoles become frogs.*
H The tadpole become the frog
E The tadpole becomes the frog.

9. *The maid arranged the knives and forks on the table.*
H The maid enumerated the the knife and the to the table.
E The maid enumerated knives and the folks to the table.

10. *A strong yen is shaking the economy.*
H The strong yen the economy.
E A strong yen is jolting the economy.

Figure 7.10: Transfer MT HSSR and ERG/ACE realization examples from the Tanaka dataset. The original sentence is shown in italics, while the two realizations are shown below (*H* - HSSR system; *E* - ERG/ACE system).

1. *There was a strong wind that day.*
   H The strong wind was blowing that day
   E A strong wind was blowing that day.

2. *History repeats itself.*
   H The history repeats
   E The history repeats you.

3. *Those ruins were once a splendid palace.*
   H This ruin the palace well katsute
   E This ruin katsutely was the palaces well.

4. *We visited Mito Park, which is famous for its plum blossoms.*
   H We did the mito, park, who was famous as the ume flower to the mi
   E We did the famous Mito parks, as the ume flower, to the mi.

5. *Appearances are against her.*
   H The situation of the hungarian national the disadvantage to her
   E The situation is the disadvantages, to her.

(a) HSSR and ERG/ACE++ realization examples which ERG/ACE could not realize. The reference sentence is shown in italics, while the two realizations are shown below (*H* - HSSR system; *E* - ERG/ACE++ system).

1. *The sound of the violin is very sweet.*
   The violin is very beautiful

2. *There are many races in the United States.*
   The many race are living in to america

3. *China's desert support more people than are in Japan.*
   More than japan, the chinese desert are supporting the many man

4. *The Sphinx began to walk around him.*
   Sphinx begin his mawari walk

5. *I got two Bs this semester.*
   The semester the 2 ryou exist of the now.

6. *I saw a house whose roof was red.*
   The red roof.

7. *Cowards die many times before their deaths.*
   The coward dies, all the way to the front.

(b) HSSR realization example which neither of the ERG/ACE systems could realize. The reference sentence is shown in italics.

Figure 7.11: Transfer MT realization examples from the Tanaka dataset produced by the HSSR and ERG/ACE++ systems.

Chapter 8

Conclusions

In this thesis I described a hybrid approach combining the knowledge in a deep hand-built grammar with a statistical machine translation approach and subsequently demonstrated its performance on the machine translation and surface realization tasks. In this chapter, I review and discuss each of the major contributions of this thesis (originally listed in Section 1.6) and consider the associated future research directions. I conclude by enumerating minor contributions and summarizing the thesis.

8.1 Novel architecture

The statistical machine translation approach proposed in this thesis comprises three parts: (1) approaches and algorithms concerned with modelling of input DMRS graphs and parallel training examples, (2) approaches and algorithms concerned with extracting a synchronous-context free grammar from a corpus of parallel examples, and (3) approaches and algorithms concerned with transforming a previously unseen DMRS graph into a target sentence using the extracted synchronous context-free grammar.

In Chapter 3, I introduced three methods for transforming DMRS graphs which remove redundant and potentially harmful information to adapt the graphs for translation and realization tasks. The methods include node and edge labelling, grammar predicate filtering, and removal of cycles in the underlying undirected graph. The methods have potential to be used in other application-specific modelling of DMRS graphs, which is validated by their inclusion in the pydmrs library released alongside the publication by Copestake et al. (2016). Additionally, I described a method for extending the source-side alignments derived during ERG parsing, and combining them with alignments obtained via statistical word alignment, in order to create alignments between the two sides of parallel training examples. As with the DMRS graph modelling methods, alignment methods have applications beyond this thesis. For instance, they were used in a sentence compression approach, which I discussed in Section 7.4.1.

In Chapter 4, I extended the SCFG formalism of Hiero (Chiang, 2005, 2007) to comprise graph-to-string rules. I introduced a rule extraction algorithm to extract graph-to-string rules from aligned DMRS graph - sentence pairs. In order to aggregate graph-to-string rules into an SCFG, I addressed the graph isomorphism problem by proposing a heuristic solution to the related problem of graph canonization. I subsequently analysed and compared the grammars extracted for the translation and realization tasks and found that

(1) the full realization grammar contains significantly more rules than the full translation grammar, (2) the majority of rules of both grammars have only been observed once, and (3) that the size of both grammars grows linearly with increasing training set size, although (4) the realization grammar growth is significantly steeper than the growth of the translation grammar.

In Chapter 5, I described a novel rule application algorithm which given an input DMRS graph finds applicable rules from the SCFG. In order to use the applicable rules to translate the input DMRS graph, I adapted the ideas behind FST decoding of de Gispert et al. (2010) for graph-to-string decoding. As with rule extraction and grammar construction approaches, the proposed rule application algorithm and graph-to-string decoder are task-agnostic. I analysed their performance on both translation and realization tasks and found that (1) the running time of the rule application algorithm increases faster than linearly with increasing graph size, but represents a minor part of the total sentence decoding time; (2) the number of applicable rules per graph node found by rule application algorithm is constant for each task, but much higher for the translation task (800 versus 400 rules per graph node), a likely consequence of the noise introduced by SMT alignment; (3) the decoder running time increases (reasonably close to) linearly with increasing graph size, but is significantly longer for the translation task.

Future work The current approach does not model punctuation directly, since punctuation is not explicitly represented in DMRS graphs. Indirectly, punctuation symbols are included in extracted rules as unaligned tokens. A potential solution could be to directly model the translation of punctuation with a dedicated synchronous context-free grammar. Alternatively, a more advanced postprocessing solution (e.g., using an LSTM) could predict target-side punctuation. The graph-to-string decoder limits inputs to at most 20 nodes in size. As observed from decoding analysis, this is a limitation of the decoder implementation and not of the proposed approach. Consequently, a more efficient decoder implementation would be able to decode larger input DMRS graphs. Finally, I made many decisions in the approach proposed in this thesis, where alternatives could be explored.

8.2 Machine translation

I applied the proposed approach to the machine translation task. The approach takes advantage of the monolingual knowledge available in the English Resource Grammar for translation by incorporating it in a statistical machine translation approach. I demonstrated the ability of the approach to be used as a large-scale practical machine translation system by evaluating it on the WMT15 English-German translation task. I compared my approach to the state-of-the-art HiFST system (Blackwood et al., 2016), implementing hierarchical phrase-based translation. The evaluation shows that my approach does not improve on the state-of-the-art Hiero system on the English-German translation task in terms of BLEU and METEOR scores and according to human judgements. The manual investigation of outputs confirmed that Hiero performs better than the proposed approach, but that the difference in quality between the two systems is perhaps not as big as suggested by the automatic evaluation metrics. For instance, the manual investigation suggested that a potential strength of my approach compared to Hiero is translation of verbs, which have an important effect on translation quality, but are not adequately quantified by the automatic evaluation metrics. It also revealed some common problems, in particular insertion of filler words and sentence structure, a likely result of missing translation rules.

Future work Although translation evaluation results are somewhat disappointing, in that the proposed approach did not improve on Hiero performance, there are several avenues for future research. Neubig and Duh (2014) reported that alignment quality had a significant effect on tree-to-string translation quality. In their experiments, a T2S system trained on alignments derived with a discriminative alignment method of Riesa and Marcu (2010) outperformed the system trained on unsupervised Giza++ alignments. The approach by Riesa and Marcu (2010) uses source and target-side syntactic information and a small amount of gold alignment data[1] to train a supervised machine learning approach (implemented in the Nile toolkit[2]). I anticipate that a similar improvement in translation quality could be achieved for my approach by using the discriminative alignment approach.[3]

The manual investigation of outputs in Section 6.3.1 showed that my approach and Hiero may have complementary strengths. Integrating the two approaches has potential to yield improvements on standalone Hiero performance. Different integration strategies could be explored, for instance on the level of hypotheses (i.e., high level integration) or combining parts of hypotheses (i.e., low level integration).[4]

Tree-to-string syntax-based SMT approaches decode a parse tree into a string (see Section 2.1.2). The extension from a single parse to a forest of parse trees improves T2S translation because it allows the translation system to override the syntactic constraints imposed by a single tree (Zhang and Chiang, 2012). Similarly, the semantic constraints of an input DMRS graph could be relaxed by considering the n-best parses for translation of a single sentence. In a first attempt, the n-best DMRS graphs could be decoded independently, combining the resulting FSA hypothesis spaces. A more complex approach would encode the n-best DMRS graphs in a packed representation, potentially as a hypergraph, which would necessitate an extension to the graph-to-string decoder presented in Chapter 5. Since there is little variation in closely ranked ERG parses, a more advanced parse selection strategy than selecting n-best parses might be required for taking full advantage of hypergraph decoding. An additional related research direction is grammar extraction from hypergraphs. Mi et al. (2008a) show that extracting rules from a packed forest representation results in substantial improvements to both tree-to-string and forest-to-string translation, by alleviating the problem of parsing errors on extracted rule quality. Hence, a similar approach might be beneficial for the approach presented in this thesis.

In recent years, the rising popularity of deep learning has sparked a new machine translation paradigm referred to as *neural machine translation (NMT).* In NMT, the machine

[1]German-English gold alignment data consisting of 300 sentences can be found here: https://user.phil-fak.uni-duesseldorf.de/ tosch/downloads.html. Credit to Thomas Schoenemann for the gold annotation and data distribution, and Eva Hasler and Lars Ahrenberg for bringing it to my attention.

[2]Available at https://github.com/jasonriesa/nile.

[3]However, in preliminary experiments not discussed in this thesis, I was unable to use the Nile toolkit on the scale of the WMT15 training data.

[4]Although not discussed in this thesis, I experimented with low-level integration, in which the Hiero CYK cells represented as FSAs were combined with HSST grid coverage cells aligning to the same span of words. Preliminary experiments did not yield as substantial improvements as desired initially, but were not conclusive.

translation system is a neural network trained in an end-to-end fashion via backpropagation. A common neural network architecture used in NMT is a multilayered Long Short-Term Memory (LSTM) encoder/decoder architecture. Some of the prominent NMT approaches include the work by Sutskever et al. (2014), Bahdanau et al. (2015), and Luong et al. (2015). In a talk at the 2014 workshop on syntax, semantics, and structure in statistical translation (Baldwin, 2014), Tim Baldwin argued that semantics should be viewed in a wider sense and that recent NMT approaches should be considered as instances of novel semantic machine translation. In this view, NMT approaches learn *semantic* representations of input sentences from their surface words. This is in contrast to the approach presented in this thesis, which relies on a notion of semantics designed by linguists. An interesting avenue of future work would explore the possibility of extending an NMT approach to use semantics in the form of a DMRS as an input instead of surface words.

8.3 Realization

In addition to machine translation, I applied the proposed approach to the MRS realization task. I considered the tactical generation and regeneration interpretations of realization. Whereas the former assumes that the meaning representation is perfect, the latter accepts that the meaning representation is potentially a flawed representation of the original sentence (or another object) because it was obtained via an imperfect process (e.g., automatic parsing). I evaluated the realization performance of the proposed approach in comparison to the established approach under both interpretations of the realization task. The tactical generation interpretation required evaluating the realization systems by realizing gold representations (see Section 7.2). The proposed approach produced realizations of high quality, but nonetheless often of worse quality than the established approach. The difference in performance was smaller, however, on realization of ERG-parsed representations (see Section 7.3). Notably, the established approach was unable to realize 7% of the input representations, whereas the proposed approach realized all of them.

In Section 7.4, I demonstrated the robustness of my approach by using it as a realization system in three applications: sentence simplification, robust parsing realization, and transfer-based machine translation. In the first application (Section 7.4.1), the proposed approach was able to realize heavily modified DMRS graphs for the sentence simplification task by producing realizations of reasonable quality. When realizing the representations obtained using robust parsing (Section 7.4.2), the proposed approach demonstrated its ability to produce high quality realizations and its robustness, by realizing all input representations. It significantly outperformed the established approach, which only realized 10-30% of input representations. Finally, I used the proposed approach as the realization component in a transfer-based machine translation system (Section 7.4.3). The proposed system was able to realize representations of varying degrees of well-formedness, into realizations of potentially degraded quality. The reported coverages and the evidence observed from analysing the realization outputs indicate that the HSSR system has good potential to be used as the realization component of the transfer-based translation system.

Future work The approach was not significantly adapted for any of the three applications enumerated above. Even better performance could be achieved if the system was tuned and/or trained for the specific task. Adapting the approach to a specific task by tuning requires a small amount of task-specific gold data. For instance, in the case of transfer-based MT realization, a few hundred pairs of transferred representations and reference translations would be required. The log-linear model parameters can be subsequently adjusted via LMERT tuning to optimize the BLEU score (see Section 5.6). A more significant task adaptation entails extracting an SCFG from the task-specific training examples and combining the task-specific SCFG with the existing SCFG (if sufficient task-specific training data exists, such combination may not be necessary). An additional feature distinguishing the task-specific rules would allow the log-linear model to better use the two sets of rules after tuning. In comparison to tuning, task-specific training requires training example pairs to be aligned. For the three tasks discussed above, alignments can be obtained via statistical word alignment (see Section 3.3.3).

Using the proposed approach for transfer-based machine translation is a particularly promising application. The high precision nature of the transfer-based machine translation system of Bond et al. (2011) means that while the system produces high quality outputs, it has a low coverage. The system is consequently complemented with a back-off SMT system, which is used when the transfer-based system does not produce an output. Replacing or complementing the existing realization pipeline component with the robust realization approach would provide a significant boost to the transfer-based end-to-end coverage. Such a solution is preferable to using a back-off SMT system, since it ties closely to the remainder of the transfer-based system and is able to take advantage of its strengths. In order to demonstrate the usefulness of robust realization approach for transfer-based machine translation, further experiments need to be conducted.

8.4 Minor contributions

Minor contributions of the thesis consist of (1) implementations of the major contributions, released as open-source libraries, and (2) original solutions to problems I solved in order to complete this thesis, which may be useful to others but that are not widely applicable. I list them below:

1. The open-source code for DMRS graph processing[5] described in Chapter 3. The pydmrs library includes (1) an interface for the pydelphin MRS to DMRS conversion, (2) DMRS preprocessing including gpred filtering, labelling, cycle removal, source token alignment, and DMRS preprocessing for HSSR Japanese-English transfer MT realization (see Section 2.4), and (3) mapping of DMRS node and edge labels to unique ids in order to facilitate further DMRS graph operations (e.g., rule extraction).

2. The open-source code for grammar estimation and decoding[6] described in Chapters 4 and 5. The library includes code implementing rule extraction, rule application, and decoding algorithms, as well as scripts for running the translation and realization system on Sun Grid Engine and Apache Spark (see Section 1.5).

[5]Located at https://github.com/matichorvat/pydmrs
[6]Located at https://github.com/matichorvat/hsst

3. The extension of the pyfst python library[7] with additional functionality needed for decoder implementation described in Chapter 5. The extensions include phi composition, encode and decode operations, a combined FST minimization and determinization operation, and enabling bypassing the use of a symbol table in favour of a more efficient operation directly with integers.

4. The DMRS modelling library[8] released alongside the Copestake et al. (2016) publication, which was a joint effort with several colleagues (listed as co-authors of the paper). The library duplicates and extends part of the functionality of the pydmrs library.[9] Notably, it also includes a browser-based DMRS graph visualization tool which is based on the work of Mike Goodman and was used to produce DMRS graph figures throughout the thesis.

5. Non-grammar rules (see Section 6.1) complement the SCFG rules extracted from a parallel corpus in order to significantly improve the robustness of the approach described in this thesis. The idea behind the functionality of non-grammar rules is not novel: disconnected graph glue rules (Section 6.1.1 are based on Hiero glue rules (Chiang, 2007), token deletion rules exist in HiFST, and various methods of transferring source words directly to target side have been used previously. However, the adaptation of these ideas to decoding of graphs and their close integration which allows them to compete with grammar rules works particularly well for the approach described in this thesis and may prove useful to others.

8.5 Summary

In this thesis I described in detail a new hybrid approach that combines the knowledge from a deep hand-built grammar such as the English Resource Grammar and a statistical machine translation approach. I demonstrated that the proposed approach is a practical approach for machine translation by applying it to a large-scale English-German translation task. I additionally demonstrated that, when applied to realization, the proposed approach provides a complementary set of strengths and weaknesses to the established MRS realization approach. In particular, it is capable of robustly realizing flawed or broken representations, making it a suitable realization equivalent to robust parsing by Zhang and Krieger (2011) and Packard and Flickinger (2017). I proposed and discussed several promising future research directions for the approach and its uses in machine translation and realization tasks. Additionally, the approach described in this thesis enables investigations into the similarities and differences between machine translation and surface realization tasks.

[7]Located at https://github.com/matichorvat/pyfst2
[8]Located at https://github.com/delph-in/pydmrs
[9]Somewhat confusingly, it also shares its name.

Bibliography

Peter Adolphs, Stephan Oepen, Ulrich Callmeier, Berthold Crysmann, Dan Flickinger, and Bernd Kiefer. Some fine points of hybrid natural language parsing. In *Proceedings of LREC*, pages 1380–1387, 2008.

Alfred V. Aho and Jeffrey D. Ullman. Syntax Directed Translations and the Pushdown Assembler. *Journal of Computer and System Sciences*, 3(1):37–56, 1969.

Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri. OpenFst: A General and Efficient Weighted Finite-State Transducer Library. In *Proceedings of the Conference on Implementation and Application of Automata*, pages 11–23, 2007.

Cyril Allauzen, Bill Byrne, Adrià de Gispert, Gonzalo Iglesias, and Michael Riley. Pushdown Automata in Statistical Machine Translation. *Computational Linguistics*, 40(3): 687–723, 2014.

Wilker Aziz, Miguel Rios, and Lucia Specia. Shallow Semantic Trees for SMT. In *Proceedings of WMT*, pages 316–322, 2011.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of ICLR*, pages 1–15, 2015.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet Project. In *Proceedings of ACL*, pages 86–90, 1998.

Timothy Baldwin. Composed, Distributed Reflections on Semantics and Statistical Machine Translation. In *Invited talk at SSST workshop*, 2014.

Timothy Baldwin, John Beavers, Emily M. Bender, Dan Flickinger, Ara Kim, and Stephan Oepen. Beauty and the beast. What running a broad-coverage precision grammar over the BNC taught us about the grammar—and the corpus. In *Linguistic evidence: empirical, theoretical, and computational perspectives*, volume 85, pages 49–70. 2005.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *Proceedings of Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, 2013.

Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, 2005.

Daniel Bauer and Owen Rambow. Hyperedge Replacement and Nonprojective Dependency Structures. In *Proceedings of the Workshop on Tree Adjoining Grammar and Related Formalism*, pages 103–111, 2016.

Leonard E Baum. An equality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. *Inequalities*, 3:1–8, 1972.

Marzieh Bazrafshan and Daniel Gildea. Semantic Roles for String to Tree Machine Translation. In *Proceedings of ACL*, pages 419–423, 2013.

Marzieh Bazrafshan and Daniel Gildea. Comparing Representations of Semantic Roles for String-To-Tree Decoding. In *Proceedings of EMNLP*, pages 1786–1791, 2014.

Anja Belz, Michael White, Dominic Espinosa, Eric Kow, Deirdre Hogan, and Amanda Stent. The First Surface Realisation Shared Task: Overview and Evaluation Results. In *Proceedings of the European Workshop on Natural Language Generation*, pages 217–226, 2011.

Anja Belz, Bernd Bohnet, Simon Mille, Leo Wanner, and Michael White. The Surface Realisation Task: Recent Developments and Future Plans. In *Proceedings of the International Natural Language Generation Conference*, pages 136–140, 2012.

Emily M Bender, Dan Flickinger, Stephan Oepen, Woodley Packard, and Ann Copestake. Layers of Interpretation: On Grammar and Compositionality. In *Proceedings of IWCS*, pages 239–249, 2015.

Oliver Bender, Evgeny Matusov, Stefan Hahn, Sasa Hasan, Shahram Khadivi, and Hermann Ney. The RWTH Arabic-to-English spoken language translation system. In *Proceedings of the Workshop on Automatic Speech Recognition & Understanding*, pages 396–401, 2007.

Graeme Blackwood, Bill Byrne, Adrià de Gispert, Federico Flego, Gonzalo Iglesias, Juan Pino, Aurelien Waite, and Tong Xiao. HiFST toolkit, 2016. URL `https://ucam-smt.github.io/`.

Ondřej Bojar and Jan Hajič. Phrase-based and deep syntactic English-to-Czech statistical machine translation. In *Proceedings of WMT*, pages 143–146, 2008.

Francis Bond, Stephan Oepen, Melanie Siegel, Ann Copestake, and Dan Flickinger. Open Source Machine Translation with DELPH-IN. In *Proceedings of Open-Source Machine Translation: Workshop at MT Summit X*, pages 15–22, 2005.

Francis Bond, Stephan Oepen, Eric Nichols, Dan Flickinger, Erik Velldal, and Petter Haugereid. Deep open-source machine translation. *Machine Translation*, 25(2):87–105, sep 2011.

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. Large language models in machine translation. In *Proceedings of EMNLP*, pages 858–867, 2007.

Fabienne Braune, Daniel Bauer, and Kevin Knight. Mapping between English Strings and Reentrant Semantic Graphs. In *Proceedings of LREC*, pages 4493–4498, 2014.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, 1990.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19:263–311, 1993.

Aoife Cahill and Josef van Genabith. Robust PCFG-Based Generation Using Automatically Acquired LFG Approximations. In *Proceedings of ACL*, pages 1033–1040, Sydney, Australia, 2006.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the Role of BLEU in Machine Translation Research. In *Proceedings of EACL*, pages 249–256, Trento, Italy, 2006.

Marine Carpuat and Dekai Wu. How Phrase Sense Disambiguation outperforms Word Sense Disambiguation for Statistical Machine Translation. In *Proceedings of TMI*, pages 43–52, 2007a.

Marine Carpuat and Dekai Wu. Improving Statistical Machine Translation using Word Sense Disambiguation. In *Proceedings of EMNLP-CoNLL*, pages 61–72, 2007b.

Marine Carpuat, Yihai Shen, Xiaofeng Yu, and Dekai Wu. Toward Integrating Word Sense and Entity Disambiguation into Statistical Machine Translation. In *Proceedings of IWSLT*, pages 37–44, 2006.

John Carroll, Ann Copestake, Dan Flickinger, and Victor Poznanski. An efficient chart generator for (semi-) lexicalist grammars. In *Proceedings of the European Workshop on Natural Language Generation*, pages 86–95, 1999.

Victor Chahuneau. pyfst, 2015. URL `http://pyfst.github.io/`.

Yee Seng Chan, Hwee Tou Ng, and David Chiang. Word Sense Disambiguation Improves Statistical Machine Translation. In *Proceedings of ACL*, pages 33–40, 2007.

Jean-Cédric Chappelier and Martin Rajman. A Generalized CYK Algorithm for Parsing Stochastic CFG. In *Proceedings of the Workshop on Tabulation in Parsing and Deduction*, pages 133–137, 1998.

Stanley F. Chen and Joshua T. Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. Technical report, Harvard University, 1998. URL `http://www.speech.sri.com/projects/srilm/manpages/pdfs/chen-goodman-tr-10-98.pdf`.

David Chiang. A Hierarchical Phrase-Based Model for Statistical Machine Translation. In *Proceedings of ACL*, pages 263–270, Michigan, USA, 2005.

David Chiang. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2): 201–228, 2007.

David Chiang. Learning to Translate with Source and Target Syntax. In *Proceedings of the ACL*, pages 1443–1452, 2010.

David Chiang and Kevin Knight. An Introduction to Synchronous Grammars. Technical report, 2006. URL `https://www3.nd.edu/{~}dchiang/papers/synchtut.pdf`.

David Chiang, Jacob Andreas, Daniel Bauer, Karl Moritz Hermann, Bevan Jones, and Kevin Knight. Parsing Graphs with Hyperedge Replacement Grammars. In *Proceedings of ACL*, pages 924–932, 2013.

Eunsol Choi, Matic Horvat, Jonathan May, Kevin Knight, and Daniel Marcu. Extracting Structured Scholarly Information from the Machine Translation Literature. In *Proceedings of LREC*, pages 421–425, 2016.

Paul Moritz Cohn. *Basic algebra.* Springer Science & Business Media, 2003.

Trevor Cohn and Mirella Lapata. Sentence Compression Beyond Word Deletion. In *Proceedings of COLING*, pages 137–144, 2008.

Ann Copestake. *Implementing Typed Feature Structure Grammars.* 2002.

Ann Copestake. Semantic Composition with (Robust) Minimal Recursion Semantics. In *Proceedings of the Workshop on Deep Linguistic Processing*, pages 73–80, 2007.

Ann Copestake. Slacker semantics: why superficiality, dependency and avoidance of commitment can be the right way to go. In *Proceedings of EACL*, pages 1–9, Athens, Greece, 2009.

Ann Copestake, Dan Flickinger, Rob Malouf, Susanne Riehemann, and Ivan Sag. Translation using Minimal Recursion Semantics. In *Proceedings of the International Conference on Theoretical and Methodological Issues in Machine Translation*, Leuven, Belgium, 1995.

Ann Copestake, Alex Lascarides, and Dan Flickinger. An Algebra for Semantic Construction in Constraint-based Grammars. In *Proceedings of ACL*, pages 140–147, 2001.

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan Sag. Minimal Recursion Semantics: An Introduction. *Journal of Research on Language and Computation*, 3(2-3): 281–332, jul 2005.

Ann Copestake, Guy Emerson, Michael Wayne Goodman, Matic Horvat, Alexander Kuhnle, and Ewa Muszyńska. Resources for building applications with Dependency Minimal Recursion Semantics. In *Proceedings of LREC*, pages 1240–1247, 2016.

Brooke Cowan, Ivona Kucerova, and Michael Collins. A Discriminative Model for Tree-to-Tree Translation. In *Proceedings of EMNLP*, pages 232–241, 2006.

Adrià de Gispert, Gonzalo Iglesias, Graeme Blackwood, Eduardo R Banga, and William Byrne. Hierarchical Phrase-Based Translation with Weighted Finite-State Transducers and Shallow-n Grammars. *Computational Linguistics*, 36(3):505–533, 2010.

Adrià de Gispert, Marcus Tomalin, and William Byrne. Word Ordering with Phrase-Based Grammars. In *Proceedings of EACL*, pages 259–268, 2014.

Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplied Data Processing on Large Clusters. *Proceedings of the Symposium on Operating Systems Design and Implementation*, pages 137–149, 2004.

Steve DeNeefe and Kevin Knight. Synchronous Tree Adjoining Machine Translation. In *Proceedings of EMNLP*, number August, page 727, 2009.

Steve DeNeefe, Kevin Knight, Wei Wang, and Daniel Marcu. What Can Syntax-based MT Learn from Phrase-based MT? In *Proceedings of EMNLP-CoNLL*, pages 755–763, 2007.

Yonggang Deng and William Byrne. MTTK: An Alignment Toolkit for Statistical Machine Translation. In *Proceedings of NAACL-HLT*, pages 265–268, New York City, USA, 2006.

Michael Denkowski and Alon Lavie. Meteor Universal: Language Specific Translation Evaluation for Any Target Language. In *Proceedings of WMT*, pages 376–380, 2014.

Yuan Ding and Martha Palmer. Machine Translation Using Probabilistic Synchronous Dependency Insertion Grammars. In *Proceedings of AAAI*, pages 541–548, 2005.

Frank Drewes, Hans-Jörg Kreowski, and Annegret Habel. Hyperedge Replacement Graph Grammars. In *Handbook of Graph Grammars and Computing by Graph Transformation 1*, pages 95–162. 1997.

Frank Drewes, Kevin Knight, and Marco Kuhlmann. Report from Dagstuhl Seminar 15122: Formal Models of Graph Transformation in Natural Language Processing. Technical Report 3, 2015.

C Dyer, A Cordova, A Mont, and J Lin. Fast, easy, and cheap: Construction of statistical machine translation models with MapReduce. In *Proceedings of WMT*, pages 199–207, 2008.

Jason Eisner. Learning non-isomorphic tree mappings for machine translation. In *Proceedings of ACL*, pages 205–208, 2003.

Dominic Espinosa, Rajakrishnan Rajkumar, Michael White, and Shoshana Berleant. Further Meta-Evaluation of Broad-Coverage Surface Realization. In *Proceedings of EMNLP*, pages 564–574, 2010.

Murhaf Fares. ERG Tokenization and Lexical Categorization. Master's thesis, University of Oslo, 2013.

Katja Filippova, Enrique Alfonseca, Carlos A Colmenares, Lukasz Kaiser, and Oriol Vinyals. Sentence Compression by Deletion with LSTMs. In *Proceedings of EMNLP*, pages 360–368, 2015.

Jeffrey Flanigan, Chris Dyer, Noah A Smith, and Jaime Carbonell. Generation from Abstract Meaning Representation using Tree Transducers. In *Proceedings of NAACL-HLT*, pages 731–739, 2016.

Dan Flickinger. On building a more effcient grammar by exploiting types. *Natural Language Engineering*, 6(1):15–28, 2000.

Dan Flickinger, Stephan Oepen, and Gisle Ytrestøl. WikiWoods: Syntacto-Semantic Annotation for English Wikipedia. In *Proceedings of LREC*, pages 1665–1671, Valletta, Malta, 2010.

Dan Flickinger, Yi Zhang, and Valia Kordoni. DeepBank: A Dynamically Annotated Treebank of the Wall Street Journal. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, pages 85–96, 2012.

Pascale Fung, Wu Zhaojun, Yang Yongsheng, and Dekai Wu. Automatic Learning of Chinese English Semantic Structure Mapping. In *Proceedings of the Spoken Language Technology Workshop*, pages 230–233, 2006.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. What's in a translation rule? In *Proceedings of NAACL-HLT*, pages 273–280, Boston, USA, 2004.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve Deneefe, Wei Wang, and Ignacio Thayer. Scalable Inference and Training of Context-Rich Syntactic Translation Models. In *Proceedings of ACL-COLING*, pages 961–968, Sydney, Australia, 2006.

Qin Gao and Stephan Vogel. Utilizing Target-Side Semantic Role Labels to Assist Hierarchical Phrase-based Machine Translation. In *Proceedings of the Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 107–115, 2011.

Michael Wayne Goodman. pyDelphin, 2016. URL `https://github.com/delph-in/pydelphin`.

Jonathan Graehl and Kevin Knight. Training Tree Transducers. In *Proceedings of NAACL-HLT*, pages 105–112, 2004.

Yvette Graham. Deep syntax in statistical machine translation. In *Proceedings of LFG12*, 2012.

Yvette Graham and Josef Van Genabith. An Open Source Rule Induction Tool for Transfer-Based SMT Yvette Graham, Josef van Genabith. *Prague Bulletin of Mathematical Linguistics*, 91:1–10, 2009.

Yvette Graham, Josef van Genabith, and Anton Bryl. F-structure transfer-based statistical machine translation. In *Proceedings of LFG09*, pages 317–337, 2009.

Jonas Groschwitz, Alexander Koller, and Christoph Teichmann. Graph parsing with s-graph grammars. In *Proceedings of ACL*, pages 1481–1490, 2015.

Rejwanul Haque, Sudip Kumar Naskar, Antal Van Den Bosch, and Andy Way. Integrating source-language context into phrase-based statistical machine translation. *Machine Translation*, 25(3):239–285, 2011.

Eva Hasler, Adrià de Gispert, Felix Stahlberg, Aurelien Waite, and Bill Byrne. Source sentence simplification for statistical machine translation. *Computer Speech & Language*, pages 1–15, 2016.

Petter Haugereid and Francis Bond. Extracting Semantic Transfer Rules from Parallel Corpora with SMT Phrase Aligners. In *Proceedings of the Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 67–75, 2012.

Aurelie Herbelot and Ann Copestake. Acquiring ontological relationships from Wikipedia using RMRS. In *Proceedings of the Workshop on Web Content*, 2006.

Matic Horvat and William Byrne. A Graph-Based Approach to String Regeneration. In *Proceedings of the Student Research Workshop at EACL*, pages 85–95, Gothenburg, Sweden, 2014.

Matic Horvat, Ann Copestake, and William Byrne. Hierarchical Statistical Semantic Realization for Minimal Recursion Semantics. In *Proceedings of IWCS*, pages 107–117, 2015.

Liang Huang and Haitao Mi. Efficient incremental decoding for tree-to-string translation. In *Proceedings of EMNLP*, pages 273–283, 2010.

Liang Huang, Kevin Knight, and Aravind Joshi. Statistical Syntax-Directed Translation with Extended Domain of locality. In *Proceedings of AMTA*, pages 66–73, 2006.

Gonzalo Iglesias, Adrià de Gispert, Eduardo R. Banga, and William Byrne. Hierarchical Phrase-Based Translation with Weighted Finite State Transducers. In *Proceedings of NAACL-HLT*, pages 433–441, Boulder, USA, 2009.

Angelina Ivanova, Stephan Oepen, Rebecca Dridan, Dan Flickinger, and Lilja Øvrelid. On Different Approaches to Syntactic Analysis Into Bi-Lexical Dependencies An Empirical Comparison of Direct, PCFG-Based, and HPSG-Based Parsers. In *Proceedings of the International Conference on Parsing Technologies*, pages 63–72, 2013.

Bevan Jones, Jacob Andreas, Daniel Bauer, Karl Moritz Hermann, and Kevin Knight. Semantics-Based Machine Translation with Hyperedge Replacement Grammars. In *Proceedings of COLING*, pages 1359–1376, 2012.

Bevan Jones, Sharon Goldwater, and Mark Johnson. Modeling Graph Languages with Grammars Extracted via Tree Decompositions. In *Proceedings of FSMNLP*, pages 54–62, 2013.

Kevin Knight and Vasileios Hatzivassiloglou. Two-level, many-paths generation. In *Proceedings of ACL*, pages 252–260, 1995.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical Phrase-Based Translation. In *Proceedings of NAACL-HLT*, pages 48–54, Edmonton, Canada, 2003.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcelo Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL: Interactive Poster and Demonstration Sessions*, pages 177–180, 2007.

Alexander Koller. Semantic construction with graph grammars. In *Proceedings of IWCS*, pages 228–238, 2015.

Alexander Koller and Marco Kuhlmann. A generalized view on parsing and translation. In *Proceedings of the International Conference on Parsing Technologies*, pages 2–13, 2011.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. Neural AMR: Sequence-to-Sequence Models for Parsing and Generation. In *arXiv preprint*, 2017.

Irene Langkilde. Forest-Based Statistical Sentence Generation. In *Proceedings of NAACL*, pages 170–177, 2000.

Irene Langkilde and Kevin Knight. Generation that Exploits Corpus-Based Statistical Knowledge. In *Proceedings of ACL*, pages 704–710, 1998.

Irene Langkilde-Geary. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Proceedings of the International Natural Language Generation Workshop*, pages 17–24, 2002.

Alon Lavie and Abhaya Agarwal. Meteor: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. In *Proceedings of WMT*, number June, pages 228–231, 2007.

Philip M. II Lewis and Richard Edwin Stearns. Syntax-directed transduction. *Journal of the ACM*, 15(3):465–488, 1968.

Junhui Li, Philip Resnik, and Hal Daumé III. Modeling Syntactic and Semantic Structures in Hierarchical Phrase-based Translation. In *Proceedings of NAACL-HLT*, pages 540–549, Atlanta, USA, 2013.

Junhui Li, Yuval Marton, Philip Resnik, and Hal Daumé III. A Unified Model for Soft Linguistic Reordering Constraints in Statistical Machine Translation. In *Proceedings of ACL*, pages 1123–1133, 2014.

Liangyou Li, Andy Way, and Qun Liu. Dependency Graph-to-String Translation. In *Proceedings of EMNLP*, pages 33–43, 2015.

Liangyou Li, Andy Way, and Qun Liu. Graph-Based Translation Via Graph Segmentation. In *Proceedings of ACL*, pages 97–107, 2016.

Dekang Lin. A path-based transfer model for machine translation. In *Proceedings of COLING*, pages 625–630, 2004.

Ding Liu and Daniel Gildea. Improved tree-to-string transducer for machine translation. In *Proceedings of WMT*, pages 62–69, 2008.

Ding Liu and Daniel Gildea. Semantic role features for machine translation. In *Proceedings of COLING*, pages 716–724, 2010.

Yang Liu, Qun Liu, and Shouxun Lin. Tree-to-String Alignment Template for Statistical Machine Translation. In *Proceedings of ACL*, pages 609–616, 2006.

Yang Liu, Yajuan Lü, and Qun Liu. Improving tree-to-tree translation with packed forests. In *Proceedings of ACL*, page 558, 2009.

Yang Liu, Qun Liu, and Yajuan Lü. Adjoining Tree-to-String Translation. In *Proceedings of ACL-HLT*, pages 1278–1287, 2011.

Jan Tore Lønning, Stephan Oepen, Dorothee Beermann, Lars Hellan, John Carroll, Helge Dyvik, Dan Flickinger, Janne Bondi Johannessen, Paul Meurer, Torbjørn Nordgard, Victoria Rosen, and Erik Velldal. LOGON - A Norwegian MT Effort. In *Proceedings of the Workshop in Recent Advances in Scandinavian Machine Translation*, Uppsala, Sweden, 2004.

Wei Lu and Hwee Tou Ng. A Probabilistic Forest-to-String Model for Language Generation from Typed Lambda Calculus Expressions. In *Proceedings of EMNLP*, pages 1611–1622, 2011.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of EMNLP*, number September, pages 1412–1421, 2015.

Wolfgang Macherey, Franz Josef Och, Ignacio Thayer, and Jakob Uszkoreit. Lattice-based Minimum Error Rate Training for Statistical Machine Translation. In *Proceedings of EMNLP*, pages 725–734, 2008.

Daniel Marcu, Wei Wang, Abdessamad Echihabi, Kevin Knight, and Marina Rey. SPMT: Statistical Machine Translation with Syntactified Target Language Phrases. In *Proceedings of EMNLP*, pages 44–52, 2006.

Mitchell P Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2): 313–330, 1993.

Haitao Mi, Liang Huang, and Qun Liu. Forest-Based Translation. In *Proceedings of ACL*, number June, pages 192–199, 2008a.

Haitao Mi, Liang Huang, and Qun Liu. Forest-Based Translation. In *Proceedings of ACL*, pages 192–199, 2008b.

Mehryar Mohri. Generic epsilon-Removal and Input epsilon-Normalization Algorithms for Weighted Transducers. *International Journal of Foundations of Computer Science*, 13(01):129–143, 2002.

Maria Nadejde, Alexandra Birch, and Philipp Koehn. Modeling Selectional Preferences of Verbs and Nouns in String-to-Tree Machine Translation. In *Proceedings of the Conference on Machine Translation*, pages 32–42, 2016.

Graham Neubig and Kevin Duh. On the Elements of an Accurate Tree-to-String Machine Translation System. In *Proceedings of ACL*, pages 143–149, 2014.

Eric Nichols, Francis Bond, Darren Scott Appling, and Yuji Matsumoto. Combining resources for open source machine translation. In *Proceedings of the Conference on Theoretical and Methodological Issues in Machine Translation*, pages 134–143, 2007.

Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of ACL*, pages 160–167, 2003.

Franz Josef Och. Statistical machine translation: foundations and recent advances. In *Proceedings of the Tenth Machine Translation Summit (MT Summit X) Tutorial*, Phuket, Thailand, 2005.

Franz Josef Och and Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51, 2003.

Stephan Oepen and Jan Tore Lønning. Discriminant-Based MRS Banking. In *Proceedings of LREC*, pages 1250–1255, 2006.

Stephan Oepen, Helge Dyvik, Jan Tore Lønning, Erik Velldal, Dorothee Beermann, John Carroll, Dan Flickinger, Lars Hellan, Janne Bondi Johannessen, Paul Meurer, Torbjørn Nordgard, and Victoria Rosen. Som å kapp-ete med trollet? -Towards MRS-Based Norwegian-English Machine Translation. In *Proceedings of the Conference on Theoretical and Methodological Issues in Machine Translation*, pages 11–20, Baltimore, USA, 2004a.

Stephan Oepen, Dan Flickinger, Kristina Toutanova, and Christopher D. Manning. LinGO Redwoods. *Research on Language and Computation*, 2(4):575–596, 2004b.

Stephan Oepen, Erik Velldal, Jan Tore Lønning, Paul Meurer, and Victoria Rosen. Towards Hybrid Quality-Oriented Machine Translation. In *Proceedings of the Conference on Theoretical and Methodological Issues in Machine Translation*, pages 144–153, 2007.

Woodley Packard. UW-MRS: Leveraging a Deep Grammar for Robotic Spatial Commands. In *Proceedings of the International Workshop on Semantic Evaluation*, pages 812–816, 2014.

Woodley Packard. Full Forest Treebanking. Master's thesis, University of Washington, 2015.

Woodley Packard. ACE parser, 2016a. URL `http://sweaglesw.org/linguistics/ace/`.

Woodley Packard. Robust Parsing in ACE, 2016b. URL `http://www.delph-in.net/2016/packard.pdf`.

Woodley Packard and Dan Flickinger. A Comparison of Robust Parsing Methods for HPSG. In *Review*, 2017.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–106, 2005.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of ACL*, pages 311–318, Philadelphia, USA, 2002.

Xiaochang Peng, Linfeng Song, and Daniel Gildea. A Synchronous Hyperedge Replacement Grammar based approach for AMR parsing. In *Proceedings of CoNLL*, pages 32–41, 2015.

Juan Pino, Aurelien Waite, and William Byrne. Simple and Efficient Model Filtering in Statistical Machine Translation. *Prague Bulletin of Mathematical Linguistics*, 98(98): 5–24, 2012.

Carl Pollard and Ivan A Sag. *Head-driven phrase structure grammar*. University of Chicago Press, 1994.

Nima Pourdamghani, Yang Gao, Ulf Hermjakob, and Kevin Knight. Aligning English Strings with Abstract Meaning Representation Graphs. In *Proceedings of EMNLP*, pages 425–429, 2014.

Nima Pourdamghani, Kevin Knight, and Ulf Hermjakob. Generating English from Abstract Meaning Representations. In *Proceedings of the International Natural Language Generation Conference*, pages 21–24, 2016.

Chris Quirk, Arul Menezes, and Colin Cherry. Dependency Treelet Translation: Syntactically Informed Phrasal SMT. In *Proceedings of ACL*, pages 271–279, 2005.

Ronald C. Read and Derek G. Corneil. The graph isomorphism disease. *Journal of Graph Theory*, 1(4):339–363, 1977.

George Rebane and Judea Pearl. The recovery of causal poly trees from statistical data. In *Proceedings of the Workshop on Uncertainty in AI*, 1987.

Jason Riesa and Daniel Marcu. Hierarchical Search for Word Alignment. In *Proceedings of ACL*, pages 157–166, 2010.

Stefan Riezler and John T. Maxwell III. Grammatical Machine Translation. In *Proceedings of the NAACL-HLT*, pages 248–255, 2006.

Libin Shen, Jinxi Xu, and Ralph Weischedel. A New String-to-Dependency Machine Translation Algorithm with a Target Dependency Language Model. In *Proceedings of ACL-HLT*, pages 577–585, 2008.

Libin Shen, Jinxi Xu, and Ralph Weischedel. String-to-dependency statistical machine translation. *Computational Linguistics*, 36(4):649–671, 2010.

Stuart M Shieber. Synchronous grammars as tree transducers. In *Proceedings of the Workshop on Tree Adjoining Grammar and Related Formalism*, pages 88–95, 2004.

Stuart M Shieber. Unifying synchronous tree-adjoining grammars and tree transducers via bimorphisms. In *Proceedings of EACL*, pages 377–384, 2006.

Stuart M Shieber. Probabilistic synchronous tree-adjoining grammars for machine translation: The argument from bilingual dictionaries. In *Proceedings of the Workshop on Syntax and Structure in Statistical Translation*, pages 88–95, 2007.

Melanie Siegel. HPSG Analysis of Japanese. In *Verbmobil: Foundations of speech-to-speech translation*, pages 264–279. Springer Berlin Heidelberg, 2000.

Melanie Siegel, Emily M. Bender, and Francis Bond. *Jacy: An Implemented Grammar of Japanese*. CSLI Publications, Stanford University, 2016.

Kiril Simov and Petya Osenova. A Hybrid Approach for Deep Machine Translation. In *Proceedings of the Deep Machine Translation Workshop*, pages 21–28, 2016.

Kiril Simov, Iliana Simova, Velislava Todorova, and Petya Osenova. Factored Models for Deep Machine Translation Bulgarian Academy of Sciences. In *Proceedings of the Deep Machine Translation Workshop*, pages 97–105, 2015.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Proceedings of NIPS*, pages 3104–3112, 2014.

Aleš Tamchyna, Chris Quirk, and Michel Galley. A Discriminative Model for Semantics-to-String Translation. In *Proceedings of the Workshop on Semantics-Drived Statistical Machine Translation*, pages 30–36, 2015.

Yasuhito Tanaka. Compilation of a multilingual parallel corpus. In *Proceedings of PACLING*, pages 265–268, 2001.

Haiqing Tang, Deyi Xiong, Min Zhang, and Zhengxian Gong. Improving Statistical Machine Translation with Selectional Preferences. In *Proceedings of COLING*, pages 2154–2163, 2016.

Lappoon R. Tang and Raymond J. Mooney. Using multiple clause constructors in inductive logic programming for semantic parsing. In *Proceedings of the European Conference on Machine Learning*, pages 466–477, 2001.

Zhaopeng Tu, Yang Liu, Young-sook Hwang Qun, and Liu Shouxun. Dependency Forest for Statistical Machine Translation. In *Proceedings of COLING*, pages 1092–1100, 2010.

Erik Velldal. *Empirical Realization Ranking*. PhD thesis, University of Oslo, 2007.

Erik Velldal and Stephan Oepen. Maximum Entropy Models for Realization Ranking. In *Proceedings of the Machine Translation Summit X*, pages 109–116, 2005.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. HMM-based Word Alignment in Statistical Machine Translation. In *Proceedings of COLING*, pages 836–841, 1996.

Aurelien Waite, Graeme Blackwood, and William Byrne. Lattice-based minimum error rate training using weighted finite-state transducers with tropical polynomial weights. In *Proceedings of FSMNLP*, pages 116–125, Blois, France, 2011.

Rui Wang, Petya Osenova, and Kiril Simov. Linguistically-Augmented Bulgarian-to-English Statistical Machine Translation Model Bulgarian Academy of Sciences. In *Proceedings of EACL*, pages 119–128, 2012.

Wei Wang, Kevin Knight, and Daniel Marcu. Binarizing Syntax Trees to Improve Syntax-Based Machine Translation Accuracy. In *Proceedings of EMNLP*, pages 746–754, 2007.

Michael White. Reining in CCG Chart Realization. In *Proceedings of the International Natural Language Generation Conference*, pages 182–191, 2004.

Michael White. Glue rules for robust chart realization. In *Proceedings of the European Workshop on Natural Language Generation*, pages 194–199, 2011.

Yuk Wah Wong and Raymond J Mooney. Generation by Inverting a Semantic Parser that Uses Statistical Machine Translation. In *Proceedings of HLT-NAACL*, pages 172–179, 2007.

Dekai Wu. Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Computational Linguistics*, 23(3):377–403, 1997.

Dekai Wu and Pascale Fung. Semantic Roles for SMT: A Hybrid Two-Pass Model. In *Proceedings of NAACL-HLT*, pages 13–16, 2009a.

Dekai Wu and Pascale Fung. Can Semantic Role Labeling Improve SMT? In *Proceedings of EAMT*, pages 218–225, 2009b.

Xianchao Wu, Takuya Matsuzaki, and Jun'ichi Tsujii. Fine-grained Tree-to-String Translation Rule Extraction. In *Proceedings of ACL*, pages 325–334, Uppsala, Sweden, 2010a.

Xianchao Wu, Takuya Matsuzaki, and Jun'Ichi Tsujii. Improve syntax-based translation using deep syntactic structures. *Machine Translation*, 24(2):141–157, 2010b.

Tong Xiao, Adrià de Gispert, Jingbo Zhu, and Bill Byrne. Effective Incorporation of Source Syntax into Hierarchical Phrase-based Translation. In *Proceedings of COLING*, pages 2064–2074, Dublin, Ireland, 2014.

Jun Xie, Haitao Mi, and Qun Liu. A novel dependency-to-string model for statistical machine translation. In *Proceedings of EMNLP*, pages 216–226, 2011.

Deyi Xiong, Qun Liu, and Shouxun Lin. A dependency treelet string correspondence model for statistical machine translation. In *Proceedings of WMT*, pages 40–47, 2007. doi:].

Deyi Xiong, Min Zhang, and Haizhou Li. Modeling the Translation of Predicate-Argument Structure for SMT. In *Proceedings of ACL*, pages 902–911, 2012.

Nianwen Xue, Jan Hajič, Martha Palmer, Zdenka Uresova, and Xiuhong Zhang. Not an Interlingua, But Close: Comparison of English AMRs to Chinese and Czech. In *Proceedings of LREC*, pages 1765–1772, 2014.

Kenji Yamada and Kevin Knight. A syntax-based statistical translation model. In *Proceedings of ACL*, pages 523–530, 2001.

Kenji Yamada and Kevin Knight. A decoder for syntax-based statistical MT. In *Proceedings of ACL*, pages 303–310, 2002.

Xiaocheng Yin, Jung-jae Kim, Zinaida Pozen, and Francis Bond. Parse Ranking with Semantic Dependencies and WordNet. In *Proceedings of Global WordNet Conference*, pages 186–193, 2014.

Gisle Ytrestøl, Dan Flickinger, and Stephan Oepen. Extracting and annotating Wikipedia sub-domains—Towards a new eScience community resource. *LOT Occasional Series*, 12:185–197, 2008.

Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark : Cluster Computing with Working Sets. In *Proceedings of the USENIX conference on Hot topics in cloud computing*, page 10, 2010.

Feifei Zhai, Jiajun Zhang, Yu Zhou, and Chengqing Zong. Machine Translation by Modeling Predicate Argument Structure Transformation. In *Proceedings of COLING*, pages 3019–3036, 2012.

Feifei Zhai, Jiajun Zhang, Yu Zhou, and Chengqing Zong. Handling Ambiguities of Bilingual Predicate-Argument Structures for Statistical Machine Translation. In *Proceedings of ACL*, pages 1127–1136, 2013.

Hui Zhang and David Chiang. An Exploration of Forest-to-String Translation: Does Translation Help or Hurt Parsing? In *Proceedings of ACL*, pages 317–321, 2012.

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim, and Sheng Li. A Tree Sequence Alignment-based Tree-to-Tree Translation Model. In *Proceedings of ACL-HLT*, pages 559–567, 2008.

Yi Zhang and Hans-Ulrich Krieger. Large-Scale Corpus-Driven PCFG Approximation of an HPSG. In *Proceedings of the Conference on Parsing Technologies*, pages 198–208, 2011.

Yue Zhang and Stephen Clark. Syntax-Based Grammaticality Improvement using CCG and Guided Search. In *Proceedings of EMNLP*, pages 1147–1157, 2011.

Appendix A

Appendix

A.1 Filtered grammar predicates

In Section 3.2.2 I describe the grammar predicate filtering procedure, in which certain grammar predicate nodes are removed from DMRS graphs. I list the thirty filtered ERG 1214 grammar predicate types below:

- approx_grad
- cop_id
- def_explicit_q
- def_implicit_q
- ellipsis
- ellipsis_expl
- ellipsis_ref
- elliptical_n
- eventuality
- focus_d
- generic_nom
- generic_verb
- hour_prep
- id
- idiom_q_i
- interval
- interval_p_end

- interval_p_start
- number_q
- parg_d
- pronoun_q
- proper_q
- property
- prpstn_to_prop
- string
- timezone_p
- udef_q
- unknown
- unspec_adj
- v_event

A.2 Translation examples

In Section 6.3.1 I discuss the performance of the HSST system on English to German translation. The discussion includes a few example translations by the system in Figure 6.7. In this section I give further 50 HSST translation examples (in particular, the first 50 sentences of the filtered newstest2014). Each example consists of three sentences: the English sentence being translated, the reference, and the 1-best HSST translation.

1. *Gutach: Increased safety for pedestrians*
   Gutach: Noch mehr Sicherheit für Fußgänger
   Gutach: Mehr Sicherheit für Fußgänger

2. *Two sets of lights so close to one another: intentional or just a silly error?*
   Zwei Anlagen so nah beieinander: Absicht oder Schildbürgerstreich?
   Dies waren die beiden two sets of lights to one another. Beabsichtigt oder nur eine dumme Fehler

3. *Yesterday, Gutacht's Mayor gave a clear answer to this question.*
   Diese Frage hat Gutachs Bürgermeister gestern klar beantwortet.
   Gestern gab der Bürgermeister eine klare Antwort auf diese Frage

4. *The Kluser lights protect cyclists, as well as those travelling by bus and the residents of Bergle.*

   Die Kluser-Ampel sichere sowohl Radfahrer als auch Busfahrgäste und die Bergle-Bewohner.

   Die Kluser Lichter, die dem Schutz der Radfahrer als auch für die Fahrt mit dem Bus und die Einwohner aus dem Bergle

5. *The system, which officially became operational yesterday, is of importance to the Sulzbachweg/Kirchstrasse junction.*

   Die gestern offiziell in Betrieb genommene Anlage sei wichtig für den Kreuzungsbereich Sulzbachweg/Kirchstraße.

   Das System, die gestern in Betrieb genommen wurden, ist von großer Bedeutung für die Kreuzung und Kirchstrasse

6. *"At times of high road and pedestrian traffic, an additional set of lights were required to ensure safety," said Eckert.*

   "Bei dem hohen Verkehrs- und Fußgängeraufkommen musste zu deren Sicherheit eine weitere Ampel her", so Eckert.

   Eckert sagte, dass eine zusätzliche Ampel mussten die für die Gewährleistung der Sicherheit in Zeiten des hohen Straße und pedestrian traffic

7. *This was also confirmed by Peter Arnold from the Offenburg District Office.*

   Dies bestätigt auch Peter Arnold vom Landratsamt Offenburg.

   Auch Peter Arnold bestätigt, dass sich das vom Bezirk Offenburg, office

8. *There are three sets of lights per direction of travel.*

   Pro Fahrtrichtung gibt es drei Lichtanlagen.

   Es gibt drei Arten von Leuchten pro Fahrtrichtung

9. *Arnold explained the technology used by the new system: It is fitted with two radar sensors.*

   Arnold erklärte die Technik der neuen Anlage: Diese ist mit zwei Radarsensoren ausgestattet.

   Arnold erklärte, die Technologie, die von dem neuen System genutzt: Es ist ausgestattet mit zwei Sensoren von Radar

10. *If the street is clear, the pedestrian obtains a green light immediately, if not, there is a delay of around 15 seconds.*

    Ist die Straße frei, kommt unmittelbar Grün für den Fußgänger, wenn nicht, dauert es etwa 15 Sekunden.

    Wenn die Straße liegt auf der Hand, sofort in der Fußgängerzone ein grünes Licht erhalten, wenn nicht, gibt es eine Verzögerung von etwa 15 Sekunden

11. *An additional radar sensor checks whether the green phase for the pedestrian can be ended.*

    Ein weiteres Radarsensor prüft, ob die Grünphase für den Fußgänger beendet werden kann.

    Ein zusätzlicher Sensor Radar wird überprüft, ob die grüne Phase für die Fußgänger zu beenden kann

12. *Of course, drivers must also play their part and keep their eyes on the road.*

    Natürlich müsse der Autofahrer hier als Partner mitdenken und die Fahrbahn beobachten.

    Natürlich müssen auch die Fahrer ihre Rolle spielen und dass sie 's eyes on the road

13. *However, Director Fresacher seems to have little trust in the text.*

    Dabei scheint Regisseur Fresacher dem Text wenig zu vertrauen.

    Allerdings Director Fresacher hat anscheinend wenig Vertrauen in den Text

14. *In particular, the actresses play a major role in the sometimes rather dubious staging.*

    Vor allem die Schauspielerinnen kommen bei den mitunter etwas fragwürdigen szenischen Umsetzungen dran.

    Vor allem die Schauspielerinnen spielen eine große Rolle in der mitunter recht fragwürdige Inszenierung

15. *They are manhandled, their heads held under water, tacked to the wall by their evening gowns.*

    Sie werden hart angefasst, mit dem Kopf unter Wasser getaucht, mit ihren Abendroben an die Wand getackert.

    Sie, die ihre Köpfe unter Wasser durch den Abend, sie an die Wand

16. *However, the source text makes barely any reference to this intense delivery.*

    Der Text vermittelt sich auf diese angestrengte Weise jedoch kaum.

    Allerdings ist der Text von Quelle ist kaum noch Bezug auf diese intensive Lieferung

17. *A black box in your car?*

    Eine Blackbox im Auto?

    Eine schwarze Box, in der Sie 's car

18. *The usually dull arena of highway planning has suddenly spawned intense debate and colorful alliances.*

    Das normalerweise eher langweilige Gebiet der Straßenplanung hat plötzlich eine intensive Debatte mit bunten Allianzen entfacht.

    Plötzlich ist die übliche langweilige Arena der Planung der Autobahn intensive Debatte und bunte Bündnisse

19. *The tea party is aghast.*

    Die Tea Party ist entsetzt.

    Die Tea Party

20. *The American Civil Liberties Union is deeply concerned, too, raising a variety of privacy issues.*

    Die amerikanische Bürgerrechtsvereinigung (ACLU) ist ebenfalls zutiefst besorgt und äußert eine Reihe von Datenschutzbedenken.

    Die American Civil Liberty Union betrifft, so tief, wodurch sich eine Vielzahl von Fragen der Privatsphäre

21. *And while Congress can't agree on whether to proceed, several states are not waiting.*

    Doch während man sich im Kongress nicht auf ein Vorgehen einigen kann, warten mehrere Bundesstaaten nicht länger.

    Und da weitergehen soll, ist erstaunlich, in dem Kongress nicht einig: Mehrere Staaten nicht warten

22. *Thousands of motorists have already taken the black boxes, some of which have GPS monitoring, for a test drive.*

    Tausende von Autofahrern haben die Fahrtenschreiber, von denen einige mit GPS-Überwachung ausgestattet sind, bereits getestet.

    Tausende Autofahrer haben sich bereits die schwarzen Boxen, die zum Teil monitoring GPS für einen test drive

23. *This really is a must for our nation.*

    Das ist wirklich ein Muss für unser Land.

    Das ist in der Tat so etwas wie ein Muss für unsere Nation

24. *There is going to be a change in how we pay these taxes.*

    Die Art und Weise, wie wir diese Steuern zahlen, wird sich verändern.

    Da wird es eine Änderung sein, wenn es darum geht, wie wir diese Steuern zahlen

25. *The technology is there to do it.*

    Die Technologie dafür ist da.

    Die Technologie ist da, um es zu tun

26. *Americans don't buy as much gas as they used to.*

    Doch in Amerika wird nicht mehr so viel getankt wie früher.

    Amerikaner kaufen keine so viel Gas, wie sie es zu

27. *Cars get many more miles to the gallon.*

    Autos verbrauchen weniger Benzin.

    Autos viel mehr Meilen pro Gallone

28. *The federal tax itself, 18.4 cents per gallon, hasn't gone up in 20 years.*

    Die staatliche Mineralölsteuer von 18,4 Cent pro Gallone (weniger als 4 Eurocent pro Liter) ist seit 20 Jahren nicht gestiegen.

    Die Steuer selbst auf Bundesebene, die 18,4 Cent pro Gallone noch nicht aus der Welt oben in 20 Jahren

29. *Politicians are loath to raise the tax even one penny when gas prices are high.*

    Politiker wagen bei hohen Spritpreisen nicht, die Steuer auch nur um einen Cent anzuheben.

    Politiker geben ungern ein, wenn die Gaspreise hoch, auf dem auch die Steuer

30. *"This works out as the most logical alternative over the long term," he said.*

    „Das stellt die langfristig sinnvollste Alternative dar“, sagte er.

    Er sagte, das Ergebnis als die vernünftigste Alternative, die sich über die lange Amtszeit

31. *Wonks call it a mileage-based user fee.*

    Bürokraten bezeichnen es als meilenbasierte Benutzergebühr.

    Wonks bezeichnet sich selbst als a user fee, die sich Meilen

32. *People are paying more directly into what they are getting.*

    Die Leute bezahlen direkt für das, was sie bekommen.

    Mehr Menschen, die dafür zahlen direkt dazu zu bringen, was sie konsumieren

33. *Several states and cities are nonetheless moving ahead on their own.*

    Mehrere Bundesstaaten und Großstädte bewegen sich nichtsdestotrotz auf eigene Faust in diese Richtung.

    Dennoch bewegt sich mehrere Staaten und Städte über ihre eigenen nächsten Schritte

34. *The most eager is Oregon, which is enlisting 5,000 drivers in the country's biggest experiment.*

    Am engagiertesten ist Oregon, das derzeit 5.000 Fahrer für das größte Experiment des Landes anwirbt.

    Die am meisten mit Spannung als 5.000 Fahrer Oregon in das Land der größte Experiment

35. *Those drivers will soon pay the mileage fees instead of gas taxes to the state.*

    Diese Fahrer werden bald die Meilengebühren statt der Mineralölsteuer an den Bundesstaat zahlen.

    Bald ist der Staat den Fahrern die Gebühren Meilen statt Gas, Steuern

36. *Nevada has already completed a pilot.*

    Nevada hat bereits ein Pilotprojekt abgeschlossen.

    Nevada hat ein Pilot bereits abgeschlossen

37. *New York City is looking into one.*

    New York City erwägt ebenfalls ein solches.

    1 als New York City.

38. *Illinois is trying it on a limited basis with trucks.*

    Illinois testet es in eingeschränktem Maße mit Lkws.

    Illinois probiert es über einen bestimmten Basis mit Lastwagen

39. *The concept is not a universal hit.*

    Das Konzept ist kein universeller Hit.

    Das Konzept ist ein universeller Hit nicht

40. *It was not something people wanted.*

    Die Leute wollten es nicht.

    Es war etwas, was Menschen nicht

41. *There is no need to build an enormous, unwieldy technological infrastructure that will inevitably be expanded to keep records of individuals' everyday comings and goings.*

    Es bestehe keine Notwendigkeit, eine gigantische, sperrige technologische Infrastruktur aufzubauen, die unweigerlich dazu verwendet werden würde, Daten über die täglichen Bewegungen von Einzelpersonen zu erfassen.

    Es gibt keine Notwendigkeit für den Bau unweigerlich eine enorme technologische Infrastruktur, um die Aufzeichnungen über die Personen aus dem täglichen Leben behalten

42. *If you can do that, Khan said, the public gets more comfortable.*

    Damit, so Khan, wäre auch die Öffentlichkeit beruhigter.

    Die Öffentlichkeit wird noch komfortabel, wenn Sie in der Lage sind, das zu tun, sagte Khan.

43. *The hunt for that technology has led some state agencies to a small California startup called True Mileage.*

    Die Jagd nach dieser Technologie hat einige Behörden zu einem kleinen Startup-Unternehmen namens True Mileage in Kalifornien geführt.

    Die Jagd, für diese Technologie ein kleines Startup namens wirklich einige staatliche Agenturen

44. *The firm was not originally in the business of helping states tax drivers.*

    Die Firma ist ursprünglich nicht angetreten, um Bundesstaaten bei der Besteuerung von Autofahrern zu helfen.

    Das Unternehmen ist nicht in das Geschäft der Staaten zu helfen, Autofahrer zu besteuern

45. *There have been some big mistakes in some of these state pilot programs.*

    In einigen dieser öffentlichen Pilotprogramme wurden große Fehler gemacht.

    In einigen dieser Staat, Pilot Programme gab es bereits einige große Fehler

46. *There are a lot less expensive and less intrusive ways to do this.*

    Es gibt wesentlich billigere und weniger intrusive Möglichkeiten, dies umzusetzen.

    Es gibt viel weniger kostspielig, weniger einschneidende Möglichkeiten, dies zu tun

47. *In Oregon, planners are experimenting with giving drivers different choices.*

    In Oregon experimentieren die Planer damit, Autofahrern eine Reihe von Auswahlmöglichkeiten zu geben.

    In Oregon Planer experimentiert mit Fahrer anders entscheiden zu geben

48. *They can choose a device with or without GPS.*

    Sie können sich für ein Gerät mit oder ohne GPS entscheiden.

    Sie können sich aussuchen, ein Gerät, mit oder ohne GPS

49. *Some transportation planners, though, wonder if all the talk about paying by the mile is just a giant distraction.*

    Einige Verkehrsplaner fragen sich allerdings, ob das ganze Gerede über das Bezahlen pro Meile nicht nur ein riesiges Ablenkungsmanöver sei.

    Einige Planer Transport muss sich fragen, ob alle, die Rede, in der es um die Bezahlung durch die Meile, nur eine riesige Ablenkung

50. *If we do this, hundreds of millions of drivers will be concerned about their privacy and a host of other things.*

    Wenn wir das tun, machen sich Hunderte Millionen von Autofahrern sorgen über ihre Privatsphäre und zahlreiche andere Dinge.

    Wenn wir das tun, Hunderte von Millionen Fahrer kümmert sich darum, ihre Privatsphäre und eine Menge anderer Dinge

A.3 Manual translation evaluation results

In Section 6.3 I report the performance of the HSST system on English to German translation compared to the HiFST system. The systems are evaluated in a manual experiment on 100 sentences by two evaluators. In this section, I show the full results of the manual evaluation. Each example consists of three sentences: the German reference sentence (italics), the HSST translation (H), and the HiFST translation (B), and the preference of each evaluator (HSST, HiFST, or neither).

1. *Jeder Hersteller plant anders.*

   H Jeder Hersteller planen sie in eine andere Art und Weise

   B Jeder Hersteller es auf eine andere Art und Weise

   Evaluator 1: neither, Evaluator 2: HSST

2. *Der öffentliche Nahverkehr werde auch teurer.*

   H Auch der öffentliche Nahverkehr wird teurer werden

   B Der öffentliche Nahverkehr wird auch teurer

   Evaluator 1: HSST, Evaluator 2: HSST

3. *Der Veganismus, er ist ein wichtiger moralischer Fingerzeig, der auf das Bewusstsein schlägt.*

   H Veganism ist ein wichtiger moralischer Kompass zu sein, das eine Wirkung auf Sie 's Bewusstsein

   B Veganismus ist eine wichtige moralische Kompass, das hat einen Effekt auf Ihr Bewusstsein

   Evaluator 1: HiFST, Evaluator 2: HiFST

4. *Soll heißen: Déjà Vu steht auch für Verlässlichkeit.*

   H Das heißt: Auch Déjà Vu steht für Zuverlässigkeit

   B Das heißt: Déjà Vu steht auch für Verlässlichkeit

   Evaluator 1: HiFST, Evaluator 2: HiFST

5. *Bei der Konzeption der frühen Internetdienste stand im Vordergrund, Kommunikation möglich zu machen.*

   H Bei der Gestaltung der frühen internet services lag der Schwerpunkt auf Kommunikation möglich machen

   B Bei der Gestaltung von den frühen Internet Services, der Schwerpunkt lag auf Kommunikation möglich

   Evaluator 1: HSST, Evaluator 2: HSST

6. *Rettungskräfte brachten das Mädchen ins Krankenhaus.*

   H Rescue workers brachte das Mädchen ins Krankenhaus

   B Rettungskräfte nahmen das Mädchen ins Krankenhaus

   Evaluator 1: HiFST, Evaluator 2: HiFST

7. *Zu uns kommen Kunden aus jeder sozialen Schicht.*

   H Wir sind der Ansicht, dass Kunden aus allen Bereichen des Lebens

   B Wir sehen Kunden aus allen Gesellschaftsschichten

   Evaluator 1: HiFST, Evaluator 2: HiFST

8. *„Nun hat Franziskus diese acht Kardinäle ausgewählt, die ihm helfen sollen“, sagte Valero.*

   H Jetzt, wo Franziskus diesen acht Kardinäle gewählt haben, um ihm zu helfen, sagte Valero

   B "Jetzt Franziskus ausgewählt hat diesen acht Kardinäle, ihm zu helfen", sagte Valero

   Evaluator 1: HSST, Evaluator 2: HSST

9. *Von großem Interesse für die Forscher sind auch Jupiter und seine Monde.*

   H Auch Jupiter und seinen Monden ist von großem Interesse für die Forscher

   B Jupiter und seinen Monden sind auch von großem Interesse für die Forscher

   Evaluator 1: neither, Evaluator 2: HiFST

10. *In einigen dieser öffentlichen Pilotprogramme wurden große Fehler gemacht.*
H In einigen dieser Staat, Pilot Programme gab es bereits einige große Fehler
B Es gab einige große Fehler in einigen dieser Staat Pilot Programme
Evaluator 1: neither, Evaluator 2: HiFST

11. *Diese Grundidee gab es aber schon seit 1913 im Ort.*
H Aber diese Grundidee, die in der Lage, die sich seit 1913
B Aber diese Grundidee gesprochen worden in den Standort seit 1913
Evaluator 1: neither, Evaluator 2: neither

12. *Was er herausfand, war schockierend.*
H Was er hat, das ist erschreckend
B Was er fand, war erschreckend
Evaluator 1: HiFST, Evaluator 2: HiFST

13. *Im Grunde genommen sind vegane Gerichte für alle da.*
H Im Grunde genommen für alle Gerichte vegan
B Im Wesentlichen, vegane Gerichte sind für jedermann
Evaluator 1: HiFST, Evaluator 2: HiFST

14. *Die Kluser-Ampel sichere sowohl Radfahrer als auch Busfahrgäste und die Bergle-Bewohner.*
H Die Kluser Lichter, die dem Schutz der Radfahrer als auch für die Fahrt mit dem Bus und die Einwohner aus dem Bergle
B Die kluser Lichter Radfahrer schützen, so gut wie die mit dem Bus zu reisen und die Bewohner von bergle
Evaluator 1: neither, Evaluator 2: neither

15. *Letztendlich entscheidet das Kind über das Geschlecht, das besser zu ihm passt – und das ist wunderbar.*
H Letztendlich wird das Kind entscheiden, ob er oder sie sich wohler fühlt, mit welcher Sex wird, und das ist eine wunderbare Sache
B Letztlich wird das Kind entscheiden, welches Geschlecht er oder sie sich wohler fühlt mit - und das ist eine wunderbare Sache
Evaluator 1: HiFST, Evaluator 2: HiFST

16. *Der in Guangzhou ansässige Neue Express hatte einen seltenen öffentlichen Aufruf zur Freilassung des Journalisten Chen Yongzhou abgedruckt.*
H Der neue Express, die sich bereit erklärt, die einem seltenen öffentlichen Plädoyer für die Freilassung von Journalist Chen Yongzhou
B Die neuen Express einen seltenen öffentlichen Plädoyer für die Freilassung des Journalisten Chen Yongzhou
Evaluator 1: neither, Evaluator 2: neither

17. *Auftakt ist bereits am Freitag die Stoppelackerparty mit DJ Ralf.*

H Das Feiern wie am Freitag mit dem Stoppelacker aus dem Bereich der Party mit DJ Ralf

B Die Feierlichkeiten beginnen am Freitag, mit dem "stoppelacker" (Feld) Party mit DJ Ralf

Evaluator 1: HiFST, Evaluator 2: HiFST

18. *Er widmet sein Leben dieser Organisation und sein Wunsch, dem kamerunischen Volk zu helfen, kennt keine Grenzen.*

H Er seinem Leben gibt, um diese Organisation, und er gewillt ist, den Menschen zu helfen

B Er gibt sein Leben für diese Organisation, und seinen Wunsch, Kamerun Menschen zu helfen ist

Evaluator 1: neither, Evaluator 2: HiFST

19. *Salem: Johanna Rahner beim Ökumenischen Gesprächsforum*

H SALEM: Johanna Rahner auf der Ecumenical Forum auf dem Laufenden

B SALEM: Johanna rahner in der ökumenischen Diskussion Forum

Evaluator 1: neither, Evaluator 2: HiFST

20. *Einige Verkehrsplaner fragen sich allerdings, ob das ganze Gerede über das Bezahlen pro Meile nicht nur ein riesiges Ablenkungsmanöver sei.*

H Einige Planer Transport muss sich fragen, ob alle, die Rede, in der es um die Bezahlung durch die Meile, nur eine riesige Ablenkung

B Einige Transport Planer, aber frage mich, ob all das Gerede über die Zahlung von der Meile ist nur eine riesige Ablenkung

Evaluator 1: neither, Evaluator 2: neither

21. *Es bestehe keine Notwendigkeit, eine gigantische, sperrige technologische Infrastruktur aufzubauen, die unweigerlich dazu verwendet werden würde, Daten über die täglichen Bewegungen von Einzelpersonen zu erfassen.*

H Es gibt keine Notwendigkeit für den Bau unweigerlich eine enorme technologische Infrastruktur, um die Aufzeichnungen über die Personen aus dem täglichen Leben behalten

B Es besteht keine Notwendigkeit für den Bau einer riesigen, schwerfällig technologische Infrastruktur, die zwangsläufig erweitert werden, um Aufzeichnungen über den Alltag der Einzelnen und

Evaluator 1: HiFST, Evaluator 2: neither

22. *Die Einsatzkräfte gingen in die Wohnung und fanden die Leiche in einem Zimmer.*

H Rescue workers in die Räumlichkeiten und stellten fest, dass der Körper in ein Schlafzimmer

B Rettungskräfte in die Räumlichkeiten und den Körper in einem Schlafzimmer gefunden

Evaluator 1: neither, Evaluator 2: HSST

23. *„Sie trommeln die ganze Nacht, damit wir wach bleiben und unsere Arbeit fortsetzen können“, sagt Bwelle.*

H Trommeln Sie die ganze Nacht, um uns wach zu halten und unsere Arbeit fortsetzen, sagte Bwelle

B "Sie sind zu schlagen Trommeln die ganze Nacht wach, um uns und unsere Arbeit fortsetzen," bwelle gesagt

Evaluator 1: neither, Evaluator 2: neither

24. *„Deutschland muss aufwachen“, erklärt Oliver Grün, Präsident des BITMi, der kleine und mittlere IT-Firmen in Deutschland vertritt.*

H Oliver Grün, Präsident des BITMi vertritt deutsche IT, kleine und mittlere Unternehmen sagen, dass Deutschland sich darüber bewußt werden müssen

B "Deutschland muss aufwachen", sagt Oliver Grün, Präsident des bitmi, das sind die kleinen und mittelgroßen deutschen Unternehmen

Evaluator 1: HiFST, Evaluator 2: HiFST

25. *Highlander-Games auf dem Kaltenhof*

H In Kaltenhof highland games

B Highland Games in kaltenhof

Evaluator 1: HSST, Evaluator 2: HiFST

26. *Deutsche Ausweise haben neben M und W künftig eine dritte Zuweisung, X für intersexuell, so das Innenministerium.*

H Nach Angaben des Innenministeriums für den deutschen Pass hat, wird eine dritte Bezeichnung, die nicht mit der M oder F - X

B Deutsche Pässe haben ein Drittel andere Bezeichnung als M oder F – X, für die, nach Angaben des Innenministeriums

Evaluator 1: neither, Evaluator 2: HSST

27. *Es gibt wesentlich billigere und weniger intrusive Möglichkeiten, dies umzusetzen.*

H Es gibt viel weniger kostspielig, weniger einschneidende Möglichkeiten, dies zu tun

B Es gibt viel kostengünstiger und weniger aufdringlich Möglichkeiten, dies zu tun

Evaluator 1: HiFST, Evaluator 2: HiFST

28. *Geschicklichkeit und Kraft müssen die Teilnehmer beim Wassereimertragen über 50 Meter unter Beweis stellen.*

H Müssen dann die Teilnehmer ihr Geschick und ihre Stärke unter Beweis stellen, indem sie mit einem Eimer Wasser, 50 Meter

B Die Teilnehmer müssen dann ihre Geschicklichkeit unter Beweis stellen und Stärke mit einem Eimer Wasser 50 Meter

Evaluator 1: neither, Evaluator 2: HSST

29. *Bei dem deutschen Gesetz geht es um die Zuweisung bei der Geburt.*
H Deutsche Gesetz geht es darum, sich bei der Geburt
B Das deutsche Recht geht es um die Zuordnung bei der Geburt
Evaluator 1: neither, Evaluator 2: HiFST

30. *Daher entspricht eine herkömmliche E-Mail eher einer offenen Postkarte als einem versiegelten Brief.*
H Aus diesem Grund ist es notwendig, dass eine herkömmliche E-Mail ist wie eine Postkarte, die täglich mehr als in einem verschlossenen Brief
B Aus diesem Grund ist eine herkömmliche E-Mail eher einer offenen Postkarte als einem versiegelten Brief
Evaluator 1: neither, Evaluator 2: HiFST

31. *Reine Pflanzenmargarine sei eine gute Alternative zu Butter, Joghurt lasse durch Sojajoghurt ersetzen.*
H Rein pflanzliche Margarine ist eine gute Alternative zu Butter, Joghurt, durch Soja Joghurt ersetzt werden
B Rein pflanzliche Margarine ist eine gute Alternative zu Butter und Joghurt ersetzt werden kann durch Soja Joghurt
Evaluator 1: neither, Evaluator 2: HiFST

32. *Vegane Ernährung erfordert pflanzliche Alternativen zu Eiern, Milch und Milchprodukten.*
H Vegane Ernährung propagiert pflanzliche Alternativen zu Eiern, Milch und Milchprodukten
B Vegane Ernährung propagiert pflanzliche Alternativen zu Eiern, Milch und Milchprodukte
Evaluator 1: HSST, Evaluator 2: HSST

33. *Der Stamm muss senkrecht abgeworfen werden, soll sich einmal überschlagen und dann gerade zum Liegen kommen.*
H Wenn die Logdatei ab und dann bis zum Ende zu liegen, der gerade sein muss
B Die Anmeldung muss geworfen werden vertikal, einmal und dann abschließend liegend geradeaus
Evaluator 1: neither, Evaluator 2: HiFST

34. *Sie weiß aber auch, dass sie zu den wenigen Menschen gehört, die sich überhaupt mit dem Thema 'Wohnen im Alter' auseinandersetzen.*
H Aber sie weiß, wie sie ist einer der wenigen Menschen tatsächlich für den Umgang mit dem Thema im Alter leben
B Aber sie ist sich bewusst, dass sie zu den wenigen Menschen, die eigentlich mit dem Thema 'Wohnen im Alter
Evaluator 1: HiFST, Evaluator 2: neither

35. *Für Fragen und Informationen steht das Tourismusbüro der Stadt Haigerloch zur Verfügung.*

H Der Tourist Office der Stadt Haigerloch, dass Sie mit Informationen zur Verfügung, für Fragen und

B Die Stadt Tourismusbüro für Fragen zur Verfügung steht und Ihnen Informationen zur Verfügung zu stellen

Evaluator 1: neither, Evaluator 2: HiFST

36. *Am engagiertesten ist Oregon, das derzeit 5.000 Fahrer für das größte Experiment des Landes anwirbt.*

H Die am meisten mit Spannung als 5.000 Fahrer Oregon in das Land der größte Experiment

B Am meisten gespannt ist Oregon, die 5.000 Fahrer in das Land der größte Experiment

Evaluator 1: HiFST, Evaluator 2: neither

37. *Eine Anfrage der FDP habe das bestätigt.*

H Bestätigt wird dies durch eine Anfrage der FDP

B Eine Anfrage der FDP bestätigt

Evaluator 1: HSST, Evaluator 2: HSST

38. *Drei Tage später wurde sie von einem Spaziergänger im Steinbruch in ihrer misslichen Lage entdeckt*

H Drei Tage, in denen sie später von a dog walker in dem Steinbruch gefangen

B Sie war drei Tage später von einem Hund Walker in dem Steinbruch gefangen

Evaluator 1: neither, Evaluator 2: HiFST

39. *In den zentralen Blöcken 12 und 13 der Südtribüne regte sich nichts.*

H Die zentrale Bestandteile der der Süden stand, Blöcke, 12 und 13, in nichts nach

B In den zentralen Blöcken des Südens, die Blöcke stehen 12 und 13, bewegte sich nichts

Evaluator 1: neither, Evaluator 2: HiFST

40. *Kinderträume werden wahr*

H Children 's dreams, der Wirklichkeit

B Die Träume der Kinder

Evaluator 1: neither, Evaluator 2: HSST

41. *Die Namen der anderen teilnehmenden Abgeordneten sollen in den kommenden Tagen veröffentlicht werden.*

H In den kommenden Tagen werden zur Herausgabe von Namen von allen teilnehmenden lawmakers

B Die Namen der anderen teilnehmenden Gesetzgeber werden in den kommenden Tagen veröffentlicht

Evaluator 1: HiFST, Evaluator 2: HiFST

42. *Google steigerte sein Angebot mehrmals und bot zuletzt 4,4 Milliarden US-Dollar.*

H Google steigerte sie ’s bid, mehrmals, die im Endeffekt mit weniger als 4,4 Milliarden Dollar

B Google erhöhte sein Angebot mehrere Male, letztlich mit so viel wie 4,4 Milliarden Dollar

Evaluator 1: neither, Evaluator 2: HiFST

43. *Sie verglich mehrere Sprichwörter mit den entsprechenden Bibelstellen und erklärte die Bedeutung.*

H Sie verglich mehrere deutsche Sprichwörter, mit der entsprechenden Verse der Bibel und erklärt die Bedeutung

B Mehrere deutsche verglichen sie mit den entsprechenden Bibel Verse und erläutert die Bedeutung

Evaluator 1: HSST, Evaluator 2: HSST

44. *Im Laufe von fünf Wochen war das Angebot um 25 Millionen Barrel gestiegen.*

H In fünf Wochen hat sich Versorgung erhöht sich durch die mehr als 25 Millionen Barrel

B In fünf Wochen Lieferungen haben sich um mehr als 25 Millionen Barrel

Evaluator 1: neither, Evaluator 2: HSST

45. *Er sagte, damit sollte eine Kontaminierung der Beweise verhindert werden, was aber „übereifrig“ und schlecht umgesetzt worden sei.*

H Er sagte, Ziel war die Belastung in Beweise zu vermeiden, aber es übereifrig und armen hingerichtet

B Er sagte, es sei zur Verhinderung der Kontaminierung von Beweisen aber war "übereifrig" und schlecht ausgeführt

Evaluator 1: HiFST, Evaluator 2: HiFST

46. *Sie kommen aus einem Umkreis von 60 Kilometern um das Dorf – und das zu Fuß.*

H Sie kommen aus 60 Kilometer rund um das Dorf, und sie kommen zu Fuß

B Sie kommen aus 60 Kilometer rund um das Dorf, und sie kommen zu Fuß zu erreichen ist

Evaluator 1: HSST, Evaluator 2: HSST

47. *Das soll ruhig so weitergehen.*

H Müssen wir komponieren bleiben und weiter für sich

B Wir müssen bleiben komponiert und zu halten

Evaluator 1: neither, Evaluator 2: neither

48. *Ich meine wirklich, er sollte in den Fußstapfen der anderen Kerle folgen.*

H Ich glaube wirklich, dass er folgt den Spuren der anderen Jungs sollten

B Ich bin wirklich der Meinung, dass er auf den Spuren von den anderen Jungs

Evaluator 1: HiFST, Evaluator 2: HiFST

49. *Auf einem Geländerundgang müssen die Kinder erraten, welche Geschichten der Wahrheit entsprechen oder doch gelogen sind.*

H Zu einem Rundgang durch die Räumlichkeiten zur Verfügung. Daran müssen sich die Kinder an die Arbeit, raus Geschichten wahr sind und which have been made up

B Auf einem Rundgang durch die Räumlichkeiten, die Kinder für die Arbeit, welche Geschichten wahr sind und welche gemacht wurden

Evaluator 1: neither, Evaluator 2: HiFST

50. *Ansonsten sollten wir erwägen, es Norwegen gleichzutun und unsere wirtschaftliche Unabhängigkeit behalten.*

H Geschieht dies nicht, sollten wir darüber nachdenken, Norwegen und Behalten wir wirtschaftliche Unabhängigkeit

B Ansonsten sollten wir darüber nachdenken, Norwegen und unsere wirtschaftliche Unabhängigkeit zu behalten

Evaluator 1: neither, Evaluator 2: neither

51. *Der Zuschauer sieht, dass das Zerlegen der Keulen weiterhin Handarbeit ist.*

H Der Betrachter fest, dass die Gelenke immer noch mit der Hand

B Der Betrachter sieht, dass die Gelenke immer noch von Hand geschnitzt

Evaluator 1: neither, Evaluator 2: neither

52. *Der Einzelhandelsausschuss der IHK Frankfurt hält das "für keine gute Idee".*

H Die Retail, Komitee, das die Frankfurter Kammer aus Industrie und Handel ist der Ansicht, dass dies nicht sein, dass es eine gute Idee

B Der Einzelhandel Ausschuss der Frankfurter Industrie- und Handelskammer ist der Ansicht, dass dies "keine gute Idee

Evaluator 1: HiFST, Evaluator 2: HiFST

53. *Auch die Strafverfolgungsbehörden und Geheimdienste haben legale Möglichkeiten, E-Mails abzufangen oder zur Kenntnis zu nehmen.*

H Auch law enforcement authorities and secret services hat rechtliche Genehmigung abgefangen oder E-Mails zur Kenntnis zu nehmen

B Strafverfolgungsbehörden und Geheimdienste haben rechtliche Genehmigung abhören oder E-Mails zur Kenntnis nehmen

Evaluator 1: HiFST, Evaluator 2: HiFST

54. *Afroamerikanische und hispanische Mädchen erreichen tendenziell eher die Pubertät als ihre weißen Altersgenossinnen, wie Untersuchungen zeigen.*

H Was in der Regel erreichen und Mädchen früher als die weißen Pendants in ihrer Pubertät, research shows

B Afroamerikanischen und hispanischen Mädchen neigen dazu, zu erreichen, die Pubertät früher als ihre weißen Pendants, Forschung zeigt

Evaluator 1: HiFST, Evaluator 2: HiFST

55. *Durch den entstehenden Brand und die Rauchentwicklung wurden zwei Menschen verletzt, einer davon konnte aber noch einen Notruf absenden.*

H Das entstandene Feuer verletzt zwei Menschen, und man konnte damit die Ausbreitung von Rauch, die ein Notfall anrufen

B Zwei Menschen wurden verletzt durch das daraus resultierende Feuer und die Ausbreitung von Rauch, aber in der Lage war, einen Notruf zu machen

Evaluator 1: HiFST, Evaluator 2: HiFST

56. *Ein Blick ins Innenleben am Opschlag 8 gibt dem Betrachter auf Anhieb ein gutes Gefühl.*

H Von Anfang an ein gutes Gefühl Malta dem Betrachter einen Blick in die internen Abläufe am Opschlag, 8

B Ein Blick in die internen Abläufe opschlag 8 gibt ein positives Gefühl, von Anfang an

Evaluator 1: HiFST, Evaluator 2: HiFST

57. *Ich lese derzeit ein furchtbar trauriges Buch.*

H Diese Tage, wo ich ein furchtbar trauriges Buch lesen

B Ich lese ein furchtbar trauriges Buch in diesen Tagen

Evaluator 1: HiFST, Evaluator 2: HiFST

58. *Deutschland ist die erste europäische Nation, in der ein drittes Geschlecht für Kinder anerkannt ist, die mit nicht eindeutigen Genitalien geboren wurden.*

H Deutschland wurde die erste europäische Nation, ein drittes Geschlecht anerkannt für Babys, mit zweideutigen Genitalien geboren

B Deutschland wurde zum ersten europäischen Nation zu erkennen, ein drittes Geschlecht für Neugeborene mit zweideutigen Genitalien

Evaluator 1: HSST, Evaluator 2: HiFST

59. *Was soll man dann tun - bei Rot fahren?*

H Sollten dem fahren, was Sie tun können, wenn wir was red

B Was sollten Sie tun - fahren durch auf Rot

Evaluator 1: neither, Evaluator 2: HiFST

60. *Sie verdienten so viel wie Lehrer und könnten gut davon leben.*

H Lehrer, die die sie verdienen, und ein gutes Leben machen kann

B Sie verdienen so viel wie Lehrer und gut leben können

Evaluator 1: HiFST, Evaluator 2: HiFST

61. *In Großbritannien lag das Plus nur bei 6%, in Frankreich bei 19%.*

H In Great Britain, Wachstum liegt bei nur sechs Prozent und in Frankreich auf der 19%

B In Großbritannien, das Wachstum liegt bei nur 6%, und in Frankreich bei 19%

Evaluator 1: HiFST, Evaluator 2: HiFST

62. *Patek ist der letzte der Bali-Bomber, der vor Gericht stand.*

H Patek ist die letzte, die Justiz zu stellen, dass die Bombenleger von Bali

B Patek ist die letzte der Bombenleger von Bali, sich Gerechtigkeit zu stellen

Evaluator 1: HiFST, Evaluator 2: HiFST

63. *Es ist perfekt, aber es lügt.*

H Es ist perfekt, aber es liegt

B Es ist perfekt, aber es liegt

Evaluator 1: neither, Evaluator 2: neither

64. *Das ist kein Kampf, zu dem man kleine Kinder zu diesem Zeitpunkt zwingen sollte.*

H Das ist eine Schlacht junge Kinder an dieser Stelle muss, bestehen nicht

B Das ist nicht ein Kampf junger Kinder sollten an dieser Stelle

Evaluator 1: neither, Evaluator 2: HiFST

65. *Während die Anteile von Yahoo-Mail zuletzt gefallen waren, konnte GMail von Google deutlich zulegen.*

H Google Mail von Google, die es geschafft haben deutlich wachsen, während es in den vergangenen sank die Zahl der Konten von Yahoo Mail

B Während die Zahl der Yahoo Mail hat in jüngster Zeit gesunken, Googles Gmail gelungen, deutlich zu wachsen

Evaluator 1: HiFST, Evaluator 2: HiFST

66. *Nach Ansicht von Professor Vallortigara kommunizieren die Hunde nicht absichtlich miteinander durch diese Bewegungen.*

H Er dachte nicht, dass absichtlich die dogs each other durch diese Bewegungen, sagte Prof

B Prof. vallortigara sagte, er denke nicht, dass die Hunde wurden absichtlich die Kommunikation untereinander durch diese Bewegungen

Evaluator 1: HiFST, Evaluator 2: HiFST

67. *Mit diesem Spiel beginnt die Saison neu für uns.*

H Die Saison beginnt für uns mit dieser Partie

B Die Saison beginnt für uns mit diesem Match

Evaluator 1: HSST, Evaluator 2: HiFST

68. *Das beginnt etwa bei den Hochzeitskleidern.*

H Das beginnt mit der Hochzeit "," Kleidung

B Das beginnt mit der Hochzeit Kleidung

Evaluator 1: HiFST, Evaluator 2: HiFST

69. *„Ich äußerte eine ganze Reihe von Dingen, die man in Erwägung ziehen sollte, und das war eines davon“, erklärte Daley dem Magazin.*

H "Auf die Frage, ob geprüft, eine ganze Menge Dinge, und das war einer von ihnen das Papier Daley

B "Ich war lautstark über die Suche in eine ganze Reihe von Dingen, und das war einer von ihnen", erzählte der Zeitung

Evaluator 1: neither, Evaluator 2: neither

70. *In Oregon experimentieren die Planer damit, Autofahrern eine Reihe von Auswahlmöglichkeiten zu geben.*

H In Oregon Planer experimentiert mit Fahrer anders entscheiden zu geben

B In Oregon, Planer experimentieren mit Fahrer geben unterschiedliche Entscheidungen

Evaluator 1: HiFST, Evaluator 2: HiFST

71. *Außerdem lobte er die Familienfreundlichkeit im Landkreis.*

H Er lobte auch die familienfreundliche Ansatz in den Bezirk

B Er lobte auch das familienfreundliche Konzept in den Bezirk

Evaluator 1: HiFST, Evaluator 2: HiFST

72. *"Immobilieninvestments bieten eine attraktive Verzinsung", sagte Ulbrich.*

H Ulbrich sagte, dass property investments bietet eine attraktive Rendite abwerfen

B "Eigentum Investitionen bieten eine attraktive Rendite abwerfen", sagte Ulbrich

Evaluator 1: neither, Evaluator 2: HiFST

73. *„Sie kommen ein besseres Angebot“, sagte er.*

H Er hat erklärt, dass Sie ein besseres Angebot bekommen werde

B "Sie gehen, um eine bessere Behandlung zu bekommen", sagte er

Evaluator 1: HSST, Evaluator 2: HSST

74. *Sollte eine Mutter besorgt sein, wenn ihrer Tochter schon im Alter von sieben oder acht Jahren Brüste und Schamhaare wachsen?*

H Sollte eine Mutter die Tochter her breast, Knospen und Schambehaarung um 7 oder 8

B Eine Mutter sollte alarmiert sein, wenn ihre Tochter zu sprießen beginnt, Brust- und Schamhaare auf 7 oder 8

Evaluator 1: neither, Evaluator 2: neither

75. *Da allerdings scheiden sich die Geister, wie Stefanie Koch vom Modehaus Kleidermüller betont.*

H Allerdings unterscheidet sich die Meinungen von im Modehaus, betont als Stefanie Koch

B Allerdings gehen die Meinungen auseinander, wie Stefanie Koch von der modehaus kleidermüller betont

Evaluator 1: HiFST, Evaluator 2: HiFST

76. *Die Stadtwerke Pfullendorf bilden das letzte Glied in dieser Kette.*

H Die kommunale Energie, Pfullendorf, Unternehmen bilden das letzte Glied in dieser Kette

B Die kommunalen Energieunternehmen bildet das letzte Glied in dieser Kette

Evaluator 1: HiFST, Evaluator 2: HiFST

77. *Haben die Roboter noch eine weitere Aufgabe?*

H Andere Aufgaben hat der Roboter

B Die Roboter haben andere Aufgaben

Evaluator 1: neither, Evaluator 2: neither

78. *Obamas Rückzieher in der Gesundheitspolitik*

H Obama 's health care "," walk back

B Obamas Gesundheitsfürsorge zu Fuß zurück

Evaluator 1: neither, Evaluator 2: neither

79. *Ein Rettungswagenteam brachte den Verletzten zur ärztlichen Behandlung ins Klinikum.*

H Ein Krankenwagen brachte den verletzten Mann zur medizinischen Behandlung in die Klinik

B Ein Krankenwagen brachte den verletzten Mann in die Klinik für die medizinische Behandlung

Evaluator 1: HSST, Evaluator 2: HSST

80. *Ob es sich um einen Bewohner handle sei noch unklar, wie ein Sprecher der Polizei in Bayreuth sagte.*

H Ein Sprecher der Polizei in Bayreuth sagte, dass noch nicht das Haus, ist unklar, ob diese Person in

B Ob diese Person war, die in dem Haus leben, ist noch unklar, sagte ein Sprecher der Polizei in Bayreuth

Evaluator 1: HiFST, Evaluator 2: HiFST

81. *Rockstar fordert einen hohen Schadensersatz von Google, da es behauptet, Googles Patentverletzung sei absichtlich, so die Anschuldigung.*

H Rockstar ist auf der Suche nach größeren Schäden, wie es behauptet, dass die Google 's patent infringement, so heißt es in der Klage gegen Google

B Rockstar ist auf der Suche nach höheren Schäden gegen Google, wie es behauptet der Patentverletzung von Google ist, so die Klage

Evaluator 1: HiFST, Evaluator 2: neither

82. *Weitere nennt der Vegetarierbund zum Weltvegantag am 1. November.*

H Weitere Beispiele für die Welt "," Vegan Day werden vom Verein zur Verfügung gestellt, die am 1. November

B Die vegetarischen Verein weitere Beispiele auf Welt veganen Tag am 1. November

Evaluator 1: HiFST, Evaluator 2: HiFST

83. *Der 67-Jährige habe schließlich den Tod seiner Mutter eingeräumt.*

H Schließlich tritt der Tod seiner Mutter wurde von den Jahren, wenn es alten als bestätigt

B Die 67 Jahre alte schließlich eingeräumt, den Tod seiner Mutter

Evaluator 1: neither, Evaluator 2: HiFST

84. *Daneben ist die Rolle des alten Fischers durch und durch eine Charakterrolle, die Janson exzellent meistert.*

H Darüber hinaus ist die Rolle des alten Fischers hervorragenden Janson ein Rolle Charakter durch und durch

B Darüber hinaus ist die Rolle des alten Fischers ist eine Figur, durch und durch, was Janson Meister ausgezeichnet

Evaluator 1: neither, Evaluator 2: neither

85. *Das sind die wirklichen europäischen Neuigkeiten: Der große, nach dem Krieg gefasste Plan zur Vereinigung Europas ist ins Stocken geraten.*

H Hier ist die echte europäische news: Schließlich der große Plan, Europa zu vereinigen, wie bei der Post "," Krieg ist ins Stocken geraten

B Hier ist die echte europäische Nachricht: Der große Plan der Nachkriegszeit zu vereinen, hat Europa schließlich ins Stocken geraten ist

Evaluator 1: HiFST, Evaluator 2: HiFST

86. *Heute mit Mitte fünfzig denkt Waltraud Ries anders darüber.*

H Jetzt geht es in der Mitte der 50er Jahre her, Waltraud Ries haben unterschiedliche Gedanken, wenn es darum geht, it

B Jetzt, in ihrem, Waltraud Ries hat andere Gedanken

Evaluator 1: neither, Evaluator 2: HiFST

87. *Die Regierung gibt Renamo die Schuld für die Zusammenstöße und beschuldigt die Organisation, Soldaten angegriffen zu haben.*

H Die Regierung die Konflikte auszulösen, wirft er vor, er Angriffe gegen Soldaten

B Die Regierung wirft für die Auslösung der Zusammenstöße und ihm vorwirft, er Soldaten anzugreifen

Evaluator 1: neither, Evaluator 2: neither

88. *Pastoren könnten als Seelsorger ganz nah bei den Menschen sein.*

H Der Minister ist in der Nähe von Menschen als Arbeitnehmer können

B Als pastoralen Arbeitnehmer, Minister können für die Menschen in der Nähe

Evaluator 1: neither, Evaluator 2: neither

89. *Doch nur wenige Besucher, die die beiden filigranen Kirchtürme bestaunen, ahnen, dass es auch unter dem Dom eine Menge zu entdecken gibt.*

H Doch nur wenige Besucher die geschmückte Kirche, zwei Türme im Klaren, dass es neben der Kathedrale, haben viel zu entdecken gibt

B Aber nur wenige Besucher, die an den beiden verzierten Kirche Türme erkennen, dass es viel zu entdecken unter der Kathedrale

Evaluator 1: HiFST, Evaluator 2: HiFST

90. *Arnold erklärte die Technik der neuen Anlage: Diese ist mit zwei Radarsensoren ausgestattet.*

H Arnold erklärte, die Technologie, die von dem neuen System genutzt: Es ist ausgestattet mit zwei Sensoren von Radar

B Arnold erklärte, die Technik des neuen Systems: Es ist ausgestattet mit zwei Radar, Sensoren

Evaluator 1: HSST, Evaluator 2: HiFST

91. *Wir entschuldigen uns bei allen Ticketinhabern für die hierdurch entstandenen Unannehmlichkeiten.*

H Wir würden uns wünschen, dass für alle Inhaber von Tickets für das daraus entstehende Unannehmlichkeiten um Entschuldigung

B Wir entschuldigen uns für alle Inhaber von Tickets für jegliche Unannehmlichkeiten verursacht hat

Evaluator 1: neither, Evaluator 2: HiFST

92. *„Wir sind sicher, dass der Händler allein gehandelt hat und dass die Angelegenheit eingedämmt ist“, erklärte Credit Suisse.*

H Credit Suisse hat gesagt, wir sind uns sicher, dass der Händler alleine gehandelt, und die Sache

B "Wir sind zuversichtlich, dass der Händler allein handelte und dass die Angelegenheit", sagte der Credit Suisse

Evaluator 1: HSST, Evaluator 2: neither

93. *"Jetzt verschwindet die Scheibe im Jumbo", kündigt er die Test-Verspeisung an.*

H Und er sagte, dass das Stück verschwinden, jetzt je nach innen in die Ankündigung seiner Übersicht dessen, das Produkt

B "Und jetzt die Scheibe im Inneren verschwinden wird", sagte er, der ankündigte, seine Kostprobe des Produkts

Evaluator 1: HiFST, Evaluator 2: HiFST

94. *Statt Burgern, Rühreiern oder Gummibären hinterherzutrauern, habe der Veganer schnell ganz neue Produkte entdeckt, die Begeisterung stieg an.*

H Die vegan muss feststellen, dass die völlig neue Produkte begeistert er wirklich schnell, und nicht als vermisst "," Burger, Rührei oder

B Anstatt fehlende Burger, Rührei oder Bären, die vegane schnell entdeckt hat völlig neue Produkte, die wirklich begeistern ihn

Evaluator 1: HiFST, Evaluator 2: HiFST

95. *„Ich möchte, dass wir unseren Zauber wiederfinden“, sagte er.*

H "Ich möchte, dass wir 's back Glücksbringer, die wir bekommen, sagte er

B "Ich möchte, dass wir, um unsere Mojo zurück", sagte er

Evaluator 1: neither, Evaluator 2: HiFST

96. *Doch seine neueste Schöpfung soll noch ambitionierter werden.*

H Doch die nächsten seiner Schöpfung, der, selbst ambitionierter

B Aber seine nächste Gründung liegt, noch ehrgeiziger zu sein

Evaluator 1: HiFST, Evaluator 2: neither

97. *Peter Lau, Spezialist des technischen Rettungsteams, sagte: „Ruby hatte ein Riesenglück.“*

H Technische Rettung, Spezialist Officer Peter Lau sagte,. Ruby hatte einen sehr glücklichen Flucht

B Fachliche und technische Rettung Officer Peter Lau sagte: "Ruby hatte sehr viel Glück entkommen

Evaluator 1: neither, Evaluator 2: HiFST

98. *Wenn ein solches Kind geboren wurde, dann rief man nicht den Psychiater, sondern den Chirurgen.*

H Als diese Kinder zur Welt kam, waren Sie doch den Psychiater nicht möglich, daß Sie ein Chirurg

B Wenn diese Kinder geboren wurden, Sie nicht rufen Sie den Psychiater, Sie haben sich dafür ausgesprochen, ein Chirurg

Evaluator 1: HiFST, Evaluator 2: HiFST

99. *Seit 2006 wurden in Mexiko mehr als 77. 000 Menschen im Zusammenhang mit der Drogenkriminalität getötet.*

H Seit dem Jahr 2006 mehr als Menschen getötet worden in Verbindung mit Drogenkriminalität in Mexiko

B Seit 2006 mehr als 77.000 Menschen getötet wurden, im Zusammenhang mit Drogenkriminalität in Mexiko

Evaluator 1: HiFST, Evaluator 2: HiFST

100. *Die Tat hatte landesweit Entsetzen ausgelöst.*
  H Das Verbrechen Entsetzen bundesweit verursacht
  B Das Verbrechen hatte bundesweit Entsetzen
  Evaluator 1: HSST, Evaluator 2: HiFST

A.4 Simplification realization examples

In Section 7.4.1 I discuss the performance of the HSSR system on realizing simplified DMRS graphs. The discussion includes a few examples of realized sentences in Figure 7.4.1. In this section I give further 25 HSSR simplified DMRS realization examples (in particular, the first 25 sentences of the filtered newstest2014). Each example consists of three sentences: the unsimplified sentence, the simplified realization by HSSR, and the simplified realization produced by token alignment (a baseline approach).

1. *Gutach: Increased safety for pedestrians*
  H (Feral): Increased safety.
  A Gutach: Increased safety

2. *Two sets of lights so close to one another: intentional or just a silly error?*
  H So, two sets of lights close to each other: ', intentional or just a silly error.
  A Two sets of lights so close to one another: Intentional or just a silly error?

3. *Yesterday, Gutacht's Mayor gave a clear answer to this question.*
  H That's mayor gave a clear answer to this question.
  A Gutacht's mayor gave a clear answer to this question.

4. *The Kluser lights protect cyclists, as well as those travelling by bus and the residents of Bergle.*
  H Cyclists are protected by the lights.
  A The kluser lights protect cyclists.

5. *The system, which officially became operational yesterday, is of importance to the Sulzbachweg/Kirchstrasse junction.*
  H The system is of importance.
  A The system is of importance.

6. *"At times of high road and pedestrian traffic, an additional set of lights were required to ensure safety," said Eckert.*
  H An additional set of lights, were required to ensure safety.
  A An additional set of lights were required to ensure safety.

7. *This was also confirmed by Peter Arnold from the Offenburg District Office.*
  H This was confirmed by peter arnold
  A This was confirmed by peter arnold.

8. *There are three sets of lights per direction of travel.*
   - *H* There are sets of lights.
   - *A* There are sets of lights.

9. *Arnold explained the technology used by the new system: It is fitted with two radar sensors.*
   - *H* Arnold explained the technology used by the new system: It is fitted with radar sensors
   - *A* Arnold explained the technology used by the new system: It is fitted with radar sensors.

10. *If the street is clear, the pedestrian obtains a green light immediately, if not, there is a delay of around 15 seconds.*
    - *H* A green light is obtained by the pedestrian
    - *A* The pedestrian obtains a green light.

11. *An additional radar sensor checks whether the green phase for the pedestrian can be ended.*
    - *H* Can to end the green phase.
    - *A* The green phase can be ended.

12. *Of course, drivers must also play their part and keep their eyes on the road.*
    - *H* Drivers must also play their part.
    - *A* Drivers must also play their part.

13. *However, Director Fresacher seems to have little trust in the text.*
    - *H* Director, seems to have trust.
    - *A* Director fresacher seems to have trust.

14. *In particular, the actresses play a major role in the sometimes rather dubious staging.*
    - *H* In particular, the actresses play a major role
    - *A* In particular, the actresses play a major role.

15. *They are manhandled, their heads held under water, tacked to the wall by their evening gowns.*
    - *H* They
    - *A* They manhandled.

16. *However, the source text makes barely any reference to this intense delivery.*
    - *H* The source text makes any reference to this intense delivery
    - *A* The source text makes any reference to this intense delivery.

17. *A black box in your car?*
    - *H* A black box in your car.
    - *A* A black box in your car?

18. *The usually dull arena of highway planning has suddenly spawned intense debate and colorful alliances.*
H The dull, arena has spawned debate, intense and colorful alliances
A The dull arena has spawned intense debate and colorful alliances.

19. *The tea party is aghast.*
H The tea party are aghast
A The tea party aghast.

20. *The American Civil Liberties Union is deeply concerned, too, raising a variety of privacy issues.*
H The american civil liberty union is concerned
A The american civil liberties union is concerned.

21. *And while Congress can't agree on whether to proceed, several states are not waiting.*
H And, while it can't be congress to agree on to proceed, several states are not waiting
A And while congress can't agree on whether to proceed, several states are not waiting.

22. *Thousands of motorists have already taken the black boxes, some of which have GPS monitoring, for a test drive.*
H Motorists, however, have taken the black boxes, some gps monitoring
A Thousands motorists have taken the black boxes some of which have gps monitoring.

23. *This really is a must for our nation.*
H This is a must
A This is a must.

24. *There is going to be a change in how we pay these taxes.*
H There's going to be a change.
A There is going to be a change.

25. *The technology is there to do it.*
H The technology is there
A The technology there.

A.5 Robust parsing realization examples

In Section 7.4.2 I discuss the performance of the HSSR system on realizing MRS representations produced by a robust parser. In this section I give 25 HSSR realizations of such representations from the filtered newstest2013 dataset. Each example consists of two sentences: the original English sentence that was parsed by the robust parser and the HSSR realization.

1. *His state recently put tracking devices on 500 cars to test out a pay-by-mile system.*
Recently, he's state tracking devices on 500 cars to test a system that, by mile

2. *The free marketeers at the Reason Foundation are also fond of having drivers pay per mile.*

   Also, the free marketeers at the reason foundation, was fond of having drivers' pay per mile

3. *In case of emergency, support is provided by the Königsfeld daytime task force.*

   In case of emergency support is provided by the daytime 'task force.

4. *anniversary of the first documented mention of the town, are drawing closer.*

   Anniversary of the first documented mention of the town is drawing closer

5. *As part of the anniversary celebrations, a number of events are planned both in Geisingen and Kirchen-Hausen.*

   A number of events are planned as part of the anniversary celebrations in Geisingen and Hausen where

6. *Council sets its sights on rail system*

   It's sights on rail system, which is set by council

7. *The possibilities are, however, limited - Cheops is a small mission with a budget of 150 million Euro.*

   However, the possibilities are limited. Cheops is a small mission, with a budget of 150 million euros.

8. *They want to find our what role the giant planet has played in the development of the solar system.*

   What they want to find's a role that the giant planet has played in the development of the solar system.

9. *The new carry-on fee is for bags in the overhead bin, so small bags under the seat will still be free.*

   The new fee is for bags, so small bags under the seat are still free in the overhead bin

10. *Frontier's new carry-on fee won't start until summer, though a date hasn't been set.*

    Though a date has not been set, new frontier's to carry on fee won't start until summer.

11. *Most haven't touched carry-on bag fees, though.*

    To carry on 'fees which have not been touched by most

12. *Frontier has gone the furthest in this area, though.*

    Frontier has gone the furthest in this area.

13. *Learning rather than unemployment: Tourism pilot project for the untrained*

    For the und was trained by learning, rather than unemployment: Tourism pilot project

14. *In October, the number of jobless fell slightly by 22, to a total of 1,307.*
22 the number of jobless to a total of 1,307 slightly in october

15. *Was given a three-year prison sentence at Liverpool Crown Court*
Was given a three-year prison sentence at liverpool crown court

16. *We will find you and put you before the courts.*
You will be found and we'll put you the courts.

17. *Town Council delighted with solid budget*
Solid town council budget

18. *Because fewer investments were made in the 2012 budgetary year than planned, the reserves also came in higher.*
Also, the reserves came in higher because fewer investments were made in the 2012 budgetary year, rather than

19. *Among the issues to be addressed was that of dual nationality, in which regard both sides have opposing ideas.*
Dual nationality in what regard is among the issues to be addressed 2 were opposing ideas sides

20. *Car driver seriously injured in accident*
Accident car driver seriously in.

21. *With its ultra-modern yet cosy café ambience, it looks like a place designed to make you feel good.*
With ultra- modern yet cosy café, it's ambience, it looks like a place designed to make you feel good

22. *Not only was he received with great applause from the audience, but was also welcomed by his sister Philippine.*
But he was received with great applause from the audience, which was also welcomed by his sister, philippine

23. *However, the Rhineland team have already been refused the licence twice.*
Twice, however, the team will be Rhineland have already been refused the license.

24. *The Court of Arbitration is the authority of last resort.*
The court of arbitration, which is the authority.

25. *Piercing beep disturbs residents*
Piercing disturbs residents.

A.5 Transfer-based MT realization examples

In Section 7.4.3 I use the HSSR system as a realization component in the Japanese-English transfer-based machine translation system. The discussion includes a few example translations by the system in Figures 7.10 and 7.11. In this section I give further 50 translation examples. Each example consists of three sentences: the Japanese sentence being translated, the English reference, and the 1-best HSSR translation.

1. 彼は泳ぎを教えてくれた
   He taught me how to swim.
   He kureru it - show the oyogi

2. あなたを見ていると私はあなたのお父さんを思い出します
   You remind me of your father.
   You exist and i see you ’

3. 昨日その手紙を書いておけばよかったのだが
   I ought to have written the letter yesterday.
   If it write that ii of the yesterday

4. サリーはケンより2つ年上です
   Sally is two years senior to Ken.
   The two more senior than ken sally

5. ここではタバコを吸ってはいけない
   You are not supposed to smoke here.
   Isn’t it, as you know, the koko the cigarette

6. その事故では君が悪いのだ
   You are to blame for the accident.
   Kind yourself as that accident

7. 私は通りにそって歩いた
   I walked along the street.
   I walked along the street

8. 私はベートーベンが今までの最高の作曲家だと思います
   I think Beethoven is as great a composer as ever lived.
   I think it made the now than ever’s ka composed no the beethoven.

9. バイオリンの音色はとても美しい
   The sound of the violin is very sweet.
   The violin is very beautiful

10. 木曜島はオーストラリア最北端ヨーク岬とニューギニア島の間のトレス海峡に位置する
Thursday Island is situated in the Torres Strait between Australia's northernmost Cape York and New Guinea.

11. 文学と哲学との関係
The relationship between literature and philosophy.
The literature and the philosophy relationship

12. 彼は真実を言っていた
He said truth.
He was telling the truth.

13. 彼女の詩をどう思いますか
What do you think of her poem?
She's poem it thinks way

14. その年はペストがはやった
The plague was about that year.
That year was caught by the plague.

15. 戦争はその国を貧乏にした
The war made the country poor.
That the war made poor province.

16. 父は私の腕をつかまえた
My father caught me by the arm.
The i's caught me. '

17. 彼は病院で気が付いた
He regained consciousness in the hospital.
The ki tsuku as the hospital himself.

18. 英語が得意な人もいれば、数学が得意な人もいる
Some are good at English, and others are good at mathematics.
The hito too

19. 警察は彼の失踪を調査している
The police are looking into his disappearance.
The investigation of the him.

20. クレジットカードを盗まれてしまったのでお電話したいのですが
I'm calling because my credit card has been stolen.
It it it it the credit it

21. 私はいやいやその仕事をした
I did the work against my will.
That work i suru iyaiya

22. 彼は今度の選挙で職に復帰するだろう
He will get his job back at the next election.
As the new year's, he returns to the position

23. 残りの夏休みは働くつもりです
I plan to work the rest of the summer.
The hataraku tsumori the nokori vacation

24. 私はよく冬にスキーにいったものだ
I used to go skiing in winter.
The mono i it went yoku in winter to the ski.

25. 彼は文句無しの巨人だ
He is altogether a giant.
The phrase 'nashi kyojin he

26. 私たちの計画はうまくいくだろう
Our plan will work out well.
Our plan goes umai

27. 彼らは工場を建設するだけの資金を持っている
They have enough capital to build a second factory.
The fund, however, have been dake motsu them the factory.

28. 口を慎んだらどうだ
You'd better hold your tongue.
Dara it way it abstained from the mouth.

29. 私の忠告がなかったら、彼は破滅していただろう
But for my advice he would have been ruined.
If it doesn't exist the advised him myself.

30. この荷物を四時まで預かってください
Could you hold these bags until four this afternoon?
This baggage azukaru it made the than the four it yourself.

31. 最善を尽くしなさい
Do your best!
Yourself, do the time.

32. 彼はたいへん想像力に富んだ作家です
He is a very imaginative writer.
The him the hungarian national tomu the imagination a lot

33. 彼の態度にはこれという変化は見られない
He's shown no appreciable change of attitude.
He's attitude, isn't it to see the changed it toiu

34. マユコは私の質問に答えられなかった
Mayuko could not answer my question.
I don't mayuko to the

35. 彼女は乱暴な運転をする人に対しては、いつも批判的だ
She is always critical of reckless drivers.
The hito that drives the criticized teki she always about violent

36. 鳥が空を高く飛んでいる
Some birds are flying high in the sky.
The sky the bird takai exist

37. 部屋はシーンとしていた
There was quiet in the room.
It suru the room with the

38. そうは思えないねえ
No, I don't think so.
It sou isn't it

39. 彼女は私の手紙を見て腹を立てた
She was displeased at my letter.
She saw the letter of me and it took offense

40. メイドはテーブルにナイフとフォークを並べた
The maid arranged the knives and forks on the table.
The maid enumerated the the knife and the to the table.

41. 驚いた事に、その子供は横浜からはるばる一人でここにやってきた
To my surprise, the child came here by himself all the way from Yokohama.
As the one that child came harubaru from yokohama to the koto to the koko

42. 彼は恐怖で青ざめた
He turned pale with fear.
He turned pale as the fear

43. ずっとその話を避けてきたくせに
You... you never wanted to talk about it.
That story to the zutto

44. 偶然そのレストランを見つけた
I found that restaurant by accident.
It found that restaurant

45. 私は朝、食べなかったから空腹だ
I am hungry because I did not eat breakfast.
The hunger i of the did not eat it

46. 狭い部屋をせいぜい広く使った
I made the best of my small room.
Hiroi seizei it used the narrow room.

47. 私は少しもお金を持っていません
I don't have any money.
Not the money i motsu sukoshi mo

48. 和子が胸をはだけて赤ん坊に乳をふくませた
Kazuko bared her breast and fed the baby.
The breast はだける, and kazuko made for the baby to build the milk.

49. 彼は会員の特典を持っている
He has the privileges of membership.
It motsu the membership he exist

50. 彼は、彼女をはいらせるためにわきに寄った
He stood aside for her to enter.
Her himself to the tame it makes to the わき