COUPLING SPEECH RECOGNITION AND RULE-BASED MACHINE TRANSLATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

SELÇUK KÖPRÜ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
COMPUTER ENGINEERING

SEPTEMBER 2008

Approval of the thesis;

"**COUPLING SPEECH RECOGNITION AND RULE-BASED MACHINE TRANSLATION**"

submitted by **Selçuk Köprü** in partial fullfillment of the requirements for the degree of **Doctor of Philosophy in Computer Engineering, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, **Graduate School of Natural and Applied Sciences**

Prof. Dr. Volkan Atalay _____
Head of Department, **Computer Engineering**

Prof. Dr. Adnan Yazıcı _____
Supervisor, **Computer Engineering Department, METU**

Examining Committee Members:

Assoc. Prof. Dr. Cem Bozşahin _____
Computer Engineering Department, METU

Prof. Dr. Adnan Yazıcı _____
Computer Engineering Department, METU

Assoc. Prof. Dr. Tolga Çiloğlu _____
Electrical and Electronics Engineering Department, METU

Assoc. Prof. Dr. Halit Oğuztüzün _____
Computer Engineering Department, METU

Asst. Prof. Dr. Çiğdem Turhan _____
Computer Engineering Department, Atılım University

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Selçuk Köprü

Signature :

ABSTRACT

COUPLING SPEECH RECOGNITION AND RULE-BASED MACHINE TRANSLATION

Köprü, Selçuk

Ph.D., Department of Computer Engineering

Supervisor: Prof. Dr. Adnan Yazıcı

September 2008, 130 pages

The objective of this thesis was to study the coupling of automatic speech recognition (ASR) systems with rule-based machine translation (MT) systems. In this thesis, a unique approach to integrating ASR with MT for speech translation (ST) tasks was proposed. The proposed approach is unique, essentially because it includes the first rule-based MT system that can process speech data in a word graph format. Compared to other rule-based MT systems, our system processes both a word graph and a stream of words. Thus, the suggested integration method of the ASR and the rule-based MT system is more detailed than a simple software engineering practice. The second reason why it is unique is because this coupling approach performed better than the first-best and N-best list techniques, which are the only other methods used to integrate an ASR with a rule-based MT system. The enhanced performance of the coupling approach was verified with experiments.

The utilization of rule-based MT systems for ST tasks is important; however, there are some unresolved issues. Most of the literature concerning coupling

systems has focused on how to integrate ASR with statistical MT rather than rule-based MT. This is because statistical MT systems can process word graphs as input, and therefore, the resolution of ambiguities can be moved to the MT component. With the new approach proposed in this thesis, this same advantage exists in rule-based MT systems. The success of such an approach could facilitate the efficient usage of rule-based systems for ST tasks.

Keywords: speech translation, machine translation, chart parsing

ÖZ

SES TANIMA VE KURAL-TABANLI OTOMATİK ÇEVİRİ SİSTEMLERİNİN ENTEGRE EDİLMESİ

Köprü, Selçuk
Doktora, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Adnan Yazıcı

Eylül 2008, 130 sayfa

Bu tezin amacı, Otomatik Ses Tanıma (OST) sistemleri ile kural tabanlı Otomatik Çeviri (OÇ) sistemlerinin bağdaştırılmasını incelemektir. Bu tezde, Ses Çevirisi (SÇ) amacıyla OST ve OÇ sistemlerini entegre eden özgün bir yaklaşım sunuyoruz. Sunulan yöntem, kelime grafiği formatındaki ses verilerini işleyebilen ilk kural tabanlı OÇ sistemini içermesi açısından özgündür. Diğer kural tabanlı OÇ sistemleri ile kıyaslamak gerekirse, bizim sistemimiz kelime dizisine ek olarak kelime grafikleri de işleyebilmektedir. Dolayısıyla, OST ve kural tabanlı OÇ arasında önerilen entegrasyon yöntemi basit bir yazılım mühendisliği uygulamasından ötedir. Bağdaştırma yöntemimizin ilk-en iyi ve N-en iyi tekniklerinden daha iyi performans gösterdiklerini de ortaya koyuyoruz. İlk-en iyi ve N-en iyi teknikleri, OST ve kural tabanlı OÇ sistemlerini bağdaştırmak için kullanılan, bizim sunduğumuz yaklaşım haricindeki tek yöntemlerdir. Argümanlarımızın doğruluğunu deneylerle de kanıtlıyoruz.

Kural tabanlı OÇ sistemlerinin SÇ işinde kullanılmasının önemli olduğunu düşünüyoruz ve bu konuda cevaplanması gereken sorular hala mevcuttur. Bağdaştırma ile ilgili literatürün çok önemli kısmı OST ile istatistiksel OÇ sistemlerinin entegrasyonu üzerinedir. Bunun sebebi, istatistiksel OÇ sistemlerinin kelime-grafiklerini girdi olarak işleyebiliyor olmasıdır. Bu şekilde, belirsizliklerin çözümlendiği yer OÇ bileşenine ötelenmektedir. Bu tezde sunduğumuz yeni yaklaşımla birlikte, aynı avantaj kural tabanlı OÇ sistemleri için de geçerli olacaktır. Bu kazanım, kural tabanlı OÇ sistemlerinin SÇ işinde etkin olarak kullanılmasını sağlayacaktır.

Anahtar Kelimeler: ses çevirisi, otomatik çeviri, çizelge ayrıştırımı

ACKNOWLEDGMENTS

First and foremost, I would like to thank my supervisor, Adnan Yazıcı. This thesis might have gone unfinished but for his encouragement. He has been a constant source of support, inspiration and common sense throughout the course of my studies.

The work described in this thesis was carried out in a joint Apptek - Teknoloji Yazılımevi project. I would like to thank all the contributors to the GTP project. I am grateful to Jude Miller who supported me continuously throughout the project and who has been a true friend to me. Nagendra Goel was kind enough to reply to my questions about speech recognition, on which I had very little knowledge at the beginning of the project. My thanks to Mirna Miller who provided the Arabic reference translations used in the experiments. I would also like to thank the owners of Apptek; Mudar Yaghi was as eager as I was about the academic outcomes of the project. I also want to express my gratitude to everyone in Apptek because of the great hospitality they showed during the many visits throughout the years.

I received valuable and constructive comments from Tolga Çiloğlu, Ayşenur Birtürk and Cem Bozşahin. At this point, I would like to pay special tribute to Cem Bozşahin who inspired me to work on machine translation during my masters thesis. I also thank to the other members of my jury; Halit Oğuztüzün and Çiğdem Turhan. Tijen Atasoy and Ayşem Karadağ, the coordinators of the METU-Academic Writing Center, helped to improve the language in this manuscript. My thanks also to Keith Hall and Brian Roark for providing the data set used in the experiments.

I am very thankful to my beloved family, my dear wife Şima and our sweet daughters Zeynep and Sude, for believing in me and supporting me during this study. I am greatly indebted to the sacrifice they made so that I could have the time to complete the work. I am also grateful to my parents Yurtal and Naciye who always encouraged me to finish the thesis.

To my family.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

CHAPTER 1

INTRODUCTION

Language is defined as "a systematic means of communicating ideas or feelings by the use of conventionalized signs, sounds, gestures, or marks having understood meanings" in Merriam-Webster (2003). From this definition, we can describe language processing in computer science as the formalization of the systematics behind languages. Language processing involves a wide variety of sub-tasks attempting to address different aspects of language. This formalization has proved to be a difficult task, considering the limited advancements gained up to now. As part of this process, the existing schema of a natural language has to be disclosed in detail, including all exceptions. The evolving nature of languages over time has made the simulation of human linguistic behavior in computers a complex task. This dynamism brings drastic changes with time; thus, how a language is spoken and written from one generation to the next will most likely be different. This may give some insight as to why there are many different languages around the world.

People who speak different languages can communicate efficiently only with

the help of a translator. Currently, human translation is expensive and it is used only if there is a need for perfect translation. Moreover, there is the speed bottleneck for human translators. When and if machine translation (MT) systems which can produce human quality translations are developed, this can take cross-linguistic communication to a different level.

Other than MT, processes in this field that aid the cross-linguistic communication are the transcription of a spoken language and speech generation from text. Automatic speech recognition (ASR) systems convert acoustic signals into a stream of words in a specific language. In the reverse direction, text to speech (TTS) systems generate acoustic signals from a sequence of words. Figure 1.1 shows an integrated solution that incorporates ASR, MT and TTS technologies. Each person who communicates using the speech-to-speech system uses his or her own native language.

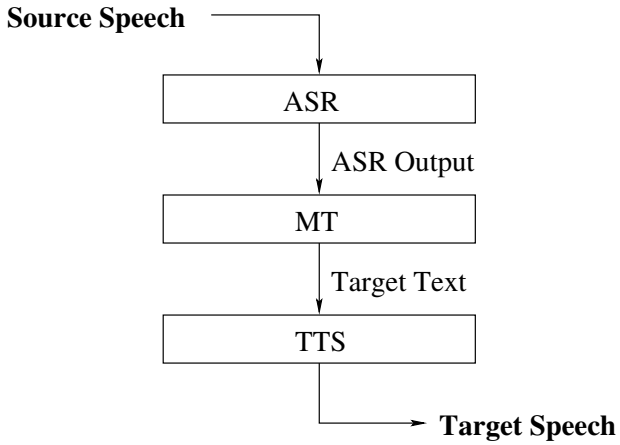

Figure 1.1: Integrated solution for speech-to-speech cross-linguistic communication.

The scenario in Figure 1.1 requires that all technologies used be as perfect as possible in order to achieve an acceptable quality of communication. Another important factor in the realization of this scenario is the integration of each module. Specifically, the integration of ASR and MT systems is significant because it can help improve the performance and quality of the overall system.

There are two basic methods that are being used to integrate ASR and rule-based MT (RBMT) systems: *First-best* method and the *N-best list* method. Both techniques are motivated from a software engineering perspective. In the first-best approach (Figure 1.2.a), the ASR module sends a single recognized text to the MT component to translate. Any ambiguity existing in the recognition process is resolved inside the ASR. In contrast to the first-best approach, in the N-best List approach (Figure 1.2.b); the ASR outputs N possible recognition hypotheses to be evaluated by the MT component. The MT picks the first hypothesis and translates it if it is grammatically correct. Otherwise, it moves to the second hypothesis and so on. If none of the available hypotheses are syntactically correct, then it translates the first one.

We propose a new method to couple ASR and rule-based MT system as an alternative to the approaches mentioned above. Figure 1.2 represents the two currently in-use coupling methods followed by the new approach we introduce (Figure 1.2.c). In the newly proposed technique, which we call the *N-best word graph* approach, the ASR module outputs a word graph containing all N-best hypotheses. The MT component parses the word graph, thus, all possible hypotheses at one time. We investigate the details and advantages of this approach

in Chapter 4.

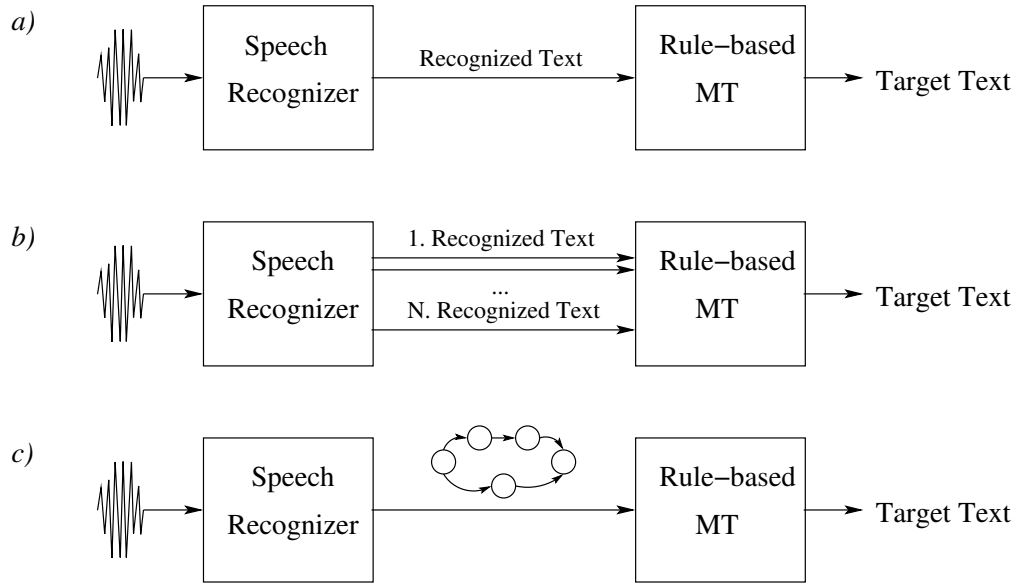

Figure 1.2: ASR and rule-based MT coupling: a) First-best b) N-best list c) N-best word graph.

There are some previous studies which address the coupling of heterogeneous components of NLP Systems (Arranz et al., 2004; Boitet & Seligman, 1994). However, our research is unique because it presents a new approach and findings in coupling statistical SR systems with a rule-based MT system. It would be very beneficial to employ rule-based MT systems for ST task because manually created linguistic resources are used extensively in RBMT. The rules and lexicons have been created over many years and are based on broad studies and experience.

Therefore, the utilization of these existing linguistic resources for both coupling and analysis would be an asset to ST.

1.1 Related Work

There is a wide range of studies available in literature that addresses the coupling issue. Harper et al. (1994) is the first study that classified coupling into three categories: tightly-coupled, loosely-coupled, and semi-coupled. According to Ringger (1995), tightness describes how closely the SR system and the syntactic parser interact with each other. In a tightly-coupled system, speech and parsing are packed into an inseparable unit. In a loosely-coupled system, processing units are contained within independent modules. Finally, semi-coupled systems lie between the previous two approaches in terms of isolation. Tight coupling is only possible if both modules are statistically-based because the unit of information interchanged between systems is meaningful for both sides. Thus, it is considered to be a difficult task to tightly couple a statistical SR with a rule-based MT. The architecture proposed in this thesis is categorized as a loosely-coupled system.

The coupling method suggested in Ney (1999) is a tightly coupled system where the whole process is based on Bayes decision rule. The work in Zhang and Kikui (2006) and Matusov et al. (2005) is similar to Ney (1999) and all are applicable only for statistical MT systems. In Saleem et al. (2004), the authors discuss another approach towards tightly coupling SR and statistical MT systems.

They conclude that using word graphs as the information exchange unit does improve performance when the weighted acoustic scores are incorporated into the MT unit. An alternative to a word graph is a *confusion network*, which is another type of directed graph where each path from start to finish includes all existing nodes. Using confusion network as the unit of exchanged information between SR and statistical MT is explored in Bertoldi and Federico (2005) and in Shen et al. (2006).

While integrating the SR system with the rule-based MT system, this study uses word graphs and chart parsing with new extensions. Parsing of word lattices[1] has been a topic of research over the past decade. The idea of chart parsing the word graph in SR systems has been previously used in different studies in order to resolve ambiguity. Tomita (1986) introduced the concept of lattice parsing for the purpose of speech recognition and used an LR parser. Next, Paeseler (1988) used a chart parser to process lattices. However, to the best of our knowledge, the specific method for chart parsing a word graph introduced in this thesis has not been previously used for coupling purposes.

Previous work on language modeling can be classified according to whether a system uses purely statistical methods or whether it uses them in combination with syntactic methods. In this thesis, the focus is on systems that contain syntactic approaches. In general, these language modeling approaches try to parse the ASR output in word graph format in order to choose the most probable hypothesis. Chow and Roukos (1989) used a unification-based Cocke-Younger-

[1] *Word graph*, *word lattice* and *lattice* are interchangeably used throughout the thesis.

Kasami (CYK) parser for the purpose of speech understanding. Chien et al. (1990a) and Weber (1994) utilized probabilistic context free grammars (PCFG) in conjunction with unification grammars to chart-parse a word lattice. There are various differences between the work of Chien et al. and Weber and the work presented in this thesis. First, in the previously mentioned studies, the chart is populated with the same word graph that comes from the speech recognizer without any pruning, whereas in our approach the word graph is reduced to an acceptable size. Otherwise, the efficiency becomes a big challenge because the search space introduced by a chart with over thousands of initial edges can easily be beyond current practical limits. We determinize and minimize the word graph using FSM algorithms before parsing it. Another important difference in our approach is the modification of the chart parsing algorithm to eliminate spurious parses. We preserve the two-dimensional structure of the word graph during parsing instead of converting it to a confusion network.

Table 1.1 summarizes the studies that work on lattice parsing. Ney (1991) deals with the use of probabilistic CYK parser for continuous speech recognition task. Stolcke (1995) summarizes extensively their approach to utilize probabilistic Earley parsing. Chappelier et al. (1999) gives an overview of different approaches to integrate linguistic models into speech recognition systems. They also research various techniques of producing sets of hypotheses that contain more "semantic" variability than the commonly used ones. Some of the recent studies about structural language modeling extract a list of N-best hypotheses using an N-gram and then apply structural methods to decide on the best hypothesis (Chelba,

Table 1.1: Summary of lattice parsing studies.

	Probabilistic	Unification-Based	Parser
Tomita (1986)			LR
Paeseler (1988)			Earley
Chow and Roukos (1989)		√	CYK
Chien et al. (1990a)	√		Active chart
Ney (1991)			CYK
Chien et al. (1993)	√	√	Active chart
Weber (1994)	√	√	Active chart
Chappelier et al. (1999)	√		CYK
Roark (2002)	√		Left-corner
Hall (2005)	√		Active chart

2000; Roark, 2001). This contrasts with the approach presented in this study where, instead of a single sentence, the word lattice is parsed. Parsing all sentence hypotheses simultaneously enables a reduction in the number of edges produced during the parsing process. This is because the shared word hypotheses are processed only once compared to the N-best list approach, where the shared words are processed each time they occur in a hypothesis. Similar to the current work, other studies parse the whole word lattice without extracting a list (Hall, 2005). A significant distinction between the work of Hall and our study is the parsing algorithm. In contrast to our chart parsing approach augmented by unification-based feature structures, Charniak parser is used in Hall's along with PCFG. Hall's results and our results are compared based on word error rate (WER) in Chapter 6.

1.2 Contributions of the thesis

The aim of this work is to contribute to the state-of-the-art in ST by employing heterogeneous components. This is accomplished through the development of a new coupling model and related algorithms. Scientific contributions achieved by this thesis are as follows:

- In this work, we introduce a new coupling approach for rule-based MT systems and ASR systems. There are some other coupling methods based on simple software engineering practices. Our approach is different from these methods because it utilizes the first rule-based MT system that can

process speech data in the form of a word graph.

- We utilize N-best word graphs produced by SR systems as an input to the MT. Ordinary unification-based chart-parsing algorithm is extended in order to handle speech lattices. The extended algorithm also eliminates spurious parse trees that occur in a confusion network.

- The approach we propose performs better than the first-best and N-best approaches. Parsing multiple hypotheses in parallel enables to drastically reduce the total number of edges in chart parsing.

- The proposed approach enables to utilize legacy rule-based MT systems effectively in speech translation task due to the ability of parsing word graph inside the MT.

- We propose a way of using statistical information in syntactic parsing which leads to the hybridization of an ordinary rule-based MT system. Statistical information is exploited by extending chart parsing, not only with unification-based feature structures, but also with functional expressions capable of modifying the feature structures.

- The experimental results obtained in this study prove that structural approaches are as competitive as statistical approaches for the language modeling task.

- The commonly-held belief that statistical MT systems are more appropriate than rule-based MT systems for speech translation task is not true anymore as a result of this thesis.

- The implementation that occurs in this thesis is more than a prototype system to be used in the experiments to prove the ideas. The developed system is a full-fledged language parser capable of processing complex grammar rules augmented with functional expression. The system is ready to be used in practical applications and it can be extended to a complete MT system.

1.3 Structure of the thesis

In the next chapter we cover the background concepts and techniques employed in this thesis. The background information contains a general review of the MT, SR, and ST concepts, along with other complementary material. In Chapter 3, we introduce our speech translation system and its components. This chapter explains the whole system from an architectural point of view. Chapter 4 focuses on the word graph pruning and word graph parsing methods. In Chapter 5, we elaborate on further enhancements of the word graph. Next, we present a set of experiments that explore the performance and quality of the proposed system. Finally, in Chapter 7 we provide a short summary of the work and conclude the thesis with future directions for research.

CHAPTER 2

BACKGROUND

In this chapter, we give a detailed overview of different material used extensively throughout the thesis.

2.1 Statistical Approaches in NLP

In natural language processing (NLP) history, structural approaches were first used for the interpretation of natural languages. Next, statistical methods were developed that claimed to perform better than their predecessors. The debate over which method is more appropriate continues because none alone could dominate the field. Meanwhile, combining these two different approaches under one system has been a significant challenge. It is now the era of hybrid systems.

Although the mathematics behind statistical modeling was defined some time ago, this approach was ignored by early computational linguists. The basic idea in statistical language modeling is to predict the solution using previously acquired knowledge. This knowledge is usually represented as the probability of

the occurrence of a possible word or word class sequence. Probabilities of word sequence occurrences can be used effectively to identify words in different NLP tasks such as SR or MT.

A very essential tool in statistical speech and language processing is the N-gram, which is a probabilistic model making use of word sequences. An N-gram model is built by looking N-1 words into the past. If the system only looks one word into the past, then that model is called a *bigram*. If it looks two words into the past then it is called a *trigram*. Probabilities in these N-grams are based on counting words in a training corpus. The success of the model is very dependent on the size of the training corpus and on the order N. High-level estimators, which encode context information in an extensive manner, perform better than low-level estimators. For example, state-of-the-art ASR systems have to use 4-grams or 5-grams. However, after a certain level, there is no contributive effect on the performance. This is exemplified in previous research, which found that 5-grams and 6-grams perform in the same manner for English ASR tasks (Jurafsky & Martin, 2008).

It is often the case that a certain word sequence is not found in a training corpus, even for low level N-Grams. So, the language model contains a zero probability value for many word sequences, which leads to poor estimates. This sparse data problem can be overcome through different *smoothing* algorithms. Good-Turing smoothing (Good, 1953) or Kneser-Ney smoothing (Kneser & Ney, 1995) are used widely to combat sparseness. The idea in smoothing is to generate a smoother distribution by discounting non-zero probability values and

incrementing the zero probability values proportionately using that discount.

Other techniques used to overcome sparseness are *back-off* and *interpolation*. In these approaches, high order N-gram values are estimated using lower-order N-grams. In interpolation, available N-gram estimates are merged to calculate the final probability, independent of the existence of a non-zero probability. In back-off, a low-order estimate is used only if the corresponding high-order estimate is zero.

There are different toolkits available for building language models. Many of them are publicly accessible for R&D purposes and they implement many useful algorithms including the ones mentioned above to overcome the sparseness problem in N-grams. The HTK toolkit (Woodland, 2000) from the University of Cambridge is portable and it is used for building and manipulating language models. Similarly, the SRILM toolkit (Stolcke, 2002) and the CMU-Cambridge toolkit (Clarkson & Rosenfeld, 1997) facilitate the construction and testing of statistical language models.

The type of corpus being used for training language models is another important facet of statistical approaches. Different language tasks require different types of training corpora. To illustrate, a corpus containing only sentences would be enough for a language generation task. Another example is part-of-speech (POS) tagging which would require a corpus with tokens tagged with appropriate word classes. Unlike these two examples, an MT task requires a parallel and aligned corpus including sentences from the source and target language.

2.2 Speech Recognition

By definition, the goal in ASR is to convert an acoustic signal to a sequence of words (Jurafsky & Martin, 2008). Figure 2.1 depicts the basic architecture of a speech recognition system. An input signal is converted into acoustic feature vectors. Analog waveforms are first digitized and quantized for further processing. Next, a sequence of extraction steps (e.g. discrete Fourier Transform) are applied to obtain the feature vectors to be used by the acoustic model. The acoustic model maps the feature vectors into symbols in order to be able to compute probabilities in decoding. A simple method of this mapping can be done using the Euclidean distance. A better approximation method to use in computing probability with acoustic vectors is to use Gaussian probability distribution functions. The Viterbi algorithm is used for the decoding process in ASR. Once decoding is complete, the most probable sequence of words is generated.

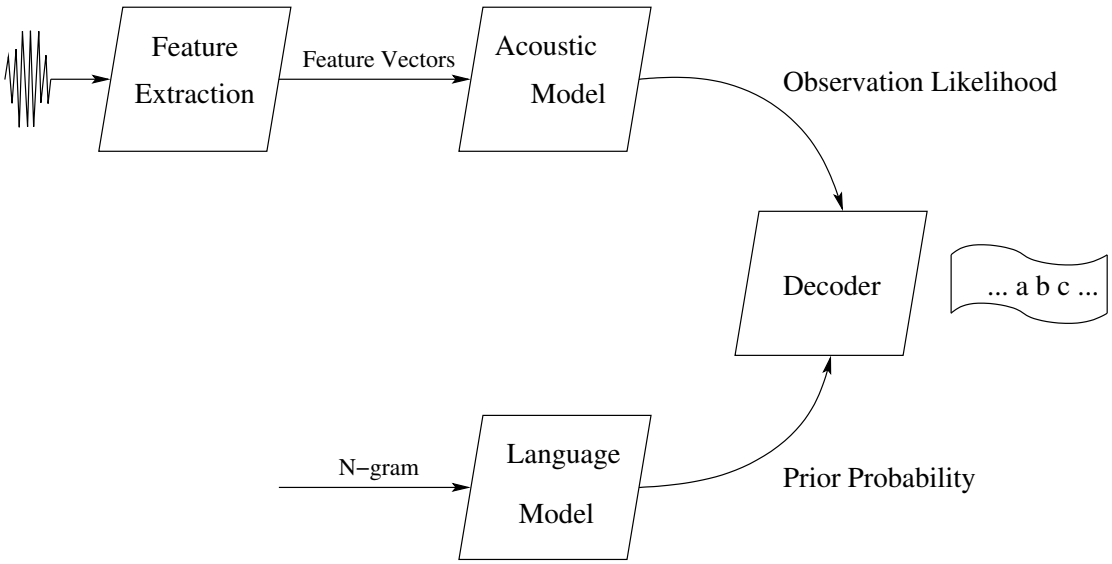

Figure 2.1: Basic ASR architecture from Jurafsky and Martin (2008).

There are certain conditions that affect the performance of an ASR system. Some of these are quality of the input signal, being noise-free or not, speaker dependency, domain, and continuity of speech. If the number of words to be recognized is low or is from a closed vocabulary, the ASR task gets simplified because the search space is reduced. In contrast, if the task is to transcribe free human conversation, then recognition becomes much harder. Another aspect of ASR involves determining whether to transcribe continuous speech or to just capture pre-specified keywords, i.e. *keyword spotting*.

In order to achieve a reasonable success rate, the system has to be trained with the same accented speech and language dialect as the target. For example, a system trained on Modern Standard Arabic (MSA) cannot be used successfully on Levantine or Iraqi dialects. Similarly, performance can drop radically while transcribing the speech of a Turkish-accented English speaker if the training corpus is a collection of native English speakers. The standard evaluation metric for ASR is the word error rate (WER).

2.2.1 Word Graphs

A word graph or a word lattice is a compact representation of multiple sentence hypotheses formed by word hypotheses. The compactness makes word graphs a popular data structure in natural language processing where ambiguity is faced in any task. A word lattice $\mathcal{L}$ is represented by a 2-tuple $< \mathcal{N}, \mathcal{A} >$ where $\mathcal{N}$ is the list of nodes representing the word hypotheses and $\mathcal{A}$ is the list of arcs representing the sequence of words. Each word hypothesis is represented by a

5-tuple $< b, e, w, a, l >$, where b is the begin time, e is the end time, w is the word, a is the acoustic likelihood of the node and l is the N-gram language model likelihood. Each arc is represented by a 3-tuple $< f, t, a >$, where f is the start node, t is the end node and a is the acoustic likelihood of the arc. A sample word graph is illustrated in Figure 2.2.

The similarity between a word-lattice and a finite state machine (FSM) enables the former to benefit from standard FSM theory and algorithms. Each node in the graph represents a unique state. Each arc connects one node to another, and is labeled with a word from within a string. Each path from the starting node to the final node describes an alternative hypothesis.

The informativeness of the FSM is increased by assigning a probability value to each label. This new form of the word graph is useful in statistical processing of natural languages. Speech recognition systems and tools use word graphs as the principal unit of output. The structure lattice format (SLF) provided by the HTK toolkit has become the standard format to describe word graphs. A word graph in SLF notation is represented by a list of nodes, a list of arcs, a start symbol and an end symbol.

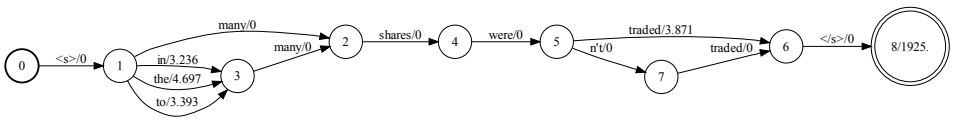

Figure 2.2: A sample word graph.

2.3 Machine Translation

Machine Translation (MT) is described as automatically producing a translated text in a target language so that the meaning remains the same as the input text in a source language (Nirenburg et al., 1994). MT research and development efforts have advanced significantly since their beginning in the 1950's; however, there is still more research to be done in order to produce human quality MT output.

Later in this section, after discussing some general issues related to machine translation, we explore the different architectural approaches. We elaborate mainly on rule-based MT systems because this thesis focuses primarily on these systems. Statistical MT is explained briefly to give a comparison to rule-based approaches. Other MT approaches such as example-based MT (EBMT) systems are not presented here because they are out of the scope of this thesis.

2.3.1 Differences in Languages

Translation can be viewed as the process of mapping structural differences between languages in addition to transferring the words from the source into the target language. The divergence between languages can be very large if they are from different families. Nevertheless, translating between languages within the same language family is a relatively simple task.

The differences in languages that contribute to the difficulty of the MT task may be various. There might be morphological differences such as those between

Turkish and Chinese; the former being highly agglutinative and the latter without any morphology at all. Thus, morphological constructions in Turkish have to be mapped to Chinese words. Similarly, syntax of languages diverges. One divergence is different word ordering in translation language pairs. For example, while translating from Arabic, which is a verb-subject-object (VSO) language, to English, a subject-verb-object (SVO) language, the required sequence manipulation has to be performed.

While doing MT, we have to face differences in any particular linguistic construction. The dissimilarity can be in a simple grammatical category as in GENDER (e.g. compare German and Turkish) or NUMBER (some languages like Arabic have DUAL marking). Moreover, the dissimilarity can also be on a grammatical description; for example, the method of relative clause construction can vary from one language to another.

2.3.2 Controlled Language

It is possible to utilize MT with a very high success rate despite the general difficulty of the task. One way of doing this is to use a controlled language as the source. Controlled language is a portion of a natural language acquired by limiting the grammar and vocabulary. In Rychtyckyj (2006), a successful implementation of a controlled language is presented. The language that is machine-translated is a restricted subset of English with a restricted grammar and lexicon. The sentences are pre-processed before they are sent to MT to check for compliance with the controlled language. In the same study, it is determined that the

utilization of MT technology in this manner results in considerable success, with high accuracy rates.

2.3.3 Rule-Based MT Approaches

The famous Vauquois triangle in Figure 2.3 shows the three mainstream approaches to MT: direct translation approach, the transfer approach and the interlingua approach. The above-mentioned figure does not include the statistical approach because the diagram pre-dates the use of statistical approaches. In direct translation, the syntactic representation of the source is not built. Rather, just a simple morphological analysis is carried out to extract word stems. No understanding of the source at the syntactic level is required. Target language constructions are built on the basis of this morphological analysis by simple word replacements and some word ordering. This approach is used in systems where the translation quality is not important and where the aim is to give the user a slight idea of what the source text is about.

The transfer approach (studied in detail in Section 2.3.4) requires a deeper level of analysis compared to the direct approach. It lies between the other two approaches in terms of complexity and extendibility. Transfer-based MT systems are composed of three main modules: analysis, transfer, and generation. In the analysis stage, morphologic and syntactic representation of the source language is constructed with the help of a linguistic grammar framework (e.g. LFG, HPSG, and CCG). Next, in the transfer stage, transfer rules are applied to the representation in order to map grammatical information from the source

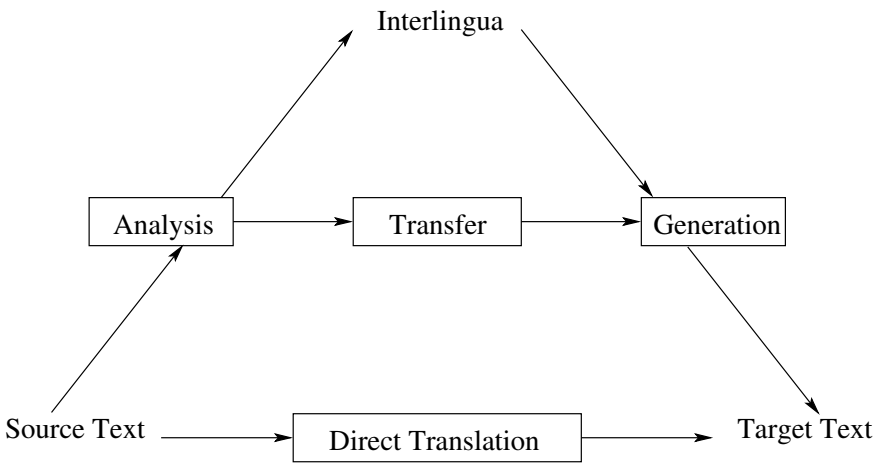

Figure 2.3: The Vauquois triangle.

language to the target language. Finally, in the generation stage, morphological generation rules and word ordering rules are executed to output the translation in the target language.

In the interlingua approach, there is no transfer module; only analysis and generation modules exist. However, this approach requires a more complex analysis and generation than the transfer approach. At the end of the analysis, the source language is converted to an interlingua, which is a language independent representation of meaning and plays the central role in this approach. The interlingua has to be rich enough to represent all possible semantic roles of constituents from different source languages. This requirement makes the analysis process much more comprehensive and forces it to go beyond syntactic analysis to include semantic analysis and real-world knowledge. In Köprü (1999), the three structural approaches are compared according to the number of necessary structural components.

2.3.4 Transfer-Based MT

In this section, we delve into transfer-based MT systems as they are used as the MT component of the proposed speech translation system. In a transfer-based MT system, the translation is obtained in three stages. Figure 2.4 represents the simplified architecture and the components used in such a system. In each stage, the system makes use of different lexicons: source lexicon, transfer lexicon, and target lexicon. The source lexicon contains lexical entries and information related to the analysis of the source language. The transfer lexicon contains the lexical translations of words from source to target language. Meaning selection rules are also placed in the transfer lexicon. The target lexicon contains lexical entries and information related to the generation of the target language.

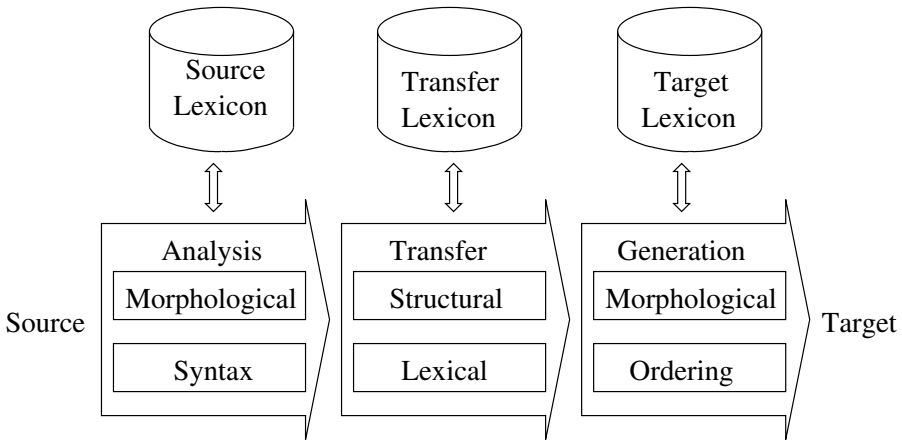

Figure 2.4: Transfer-Based MT Architecture.

In the analysis stage, morphological analysis and syntactic analysis (explored in detail in Section 2.5) are carried out. During morphological analysis, inflected

forms of the words in the source sentence are stemmed and looked up in the source lexicon. The syntactic analyzer parses the string of words to catch all linguistic information existing in the source sentence. At the end of parsing, an intermediate representation specific to the source language is constructed. This intermediate description is usually a complex data structure that facilitates the representation of the grammatical features and values required during the processing. It must be capable of representing both simple values and complex values such as lists and sets. Detailed examples of these structures are given in Section 3.3 and Section 3.4.

In the transfer stage, the source-language-specific information is mapped to a target language representation. This mapping is accomplished in two dimensions. First, the intermediate representation is modified structurally. This modification is dictated by the transfer rules according to the target language specifications. Second, word forms in the source language are replaced with appropriate meanings in the target language. Meaning selection rules that investigate the internal representation can be utilized at this stage to pick the most proper meaning.

At the final stage of a transfer-based MT system, the output in the target language is generated. Generational morphology rules are applied to the stem forms in the intermediate representation in order to synthesize the inflected word forms. During this morphological processing, information contained in the target lexicon can be utilized. Once the eventual word forms are ready, word-ordering rules are used to generate the ultimate translation.

2.3.5 Statistical MT

Statistical approaches to build the MT have been utilized since MT was first introduced. However, statistical approaches were abandoned quickly because their capabilities were underestimated. After they proved to be useful in similar language-processing tasks, such as ASR, researchers started to utilize them again for MT in the beginning of 1990s (Brown et al., 1990).

Statistical MT is very different from rule-based MT approaches. Analogous to other statistical applications like ASR, it is based on N-gram language models. The similarity in the tools being used in statistical MT and ASR leads to efficient coupling of these two components.

Given a sentence S in a source language to be translated to a sentence T in a target language, the aim is to maximize the probability $P(T|S)$. The sentence with the highest probability $\hat{T}$ is:

$$\hat{T} = \underset{T}{argmax}\ P(T|S) \tag{2.1}$$

$$\hat{T} = \underset{T}{argmax}\ P(S|T)P(T) \tag{2.2}$$

The fundamental equation of statistical MT (2.2) includes the following two components: $P(S|T)$, the translation model, and $P(T)$, the language model of the target language. The language model component keeps track of the fluency of the generated text, and the translation model keeps track of the faithfulness of the translation (Jurafsky & Martin, 2008).

In Och (2002), different advantages of the statistical MT approach over the other rule-based approaches are listed. The main advantage is the relatively smaller amount of time and human resources required to develop an MT for a new language pair.

2.3.6 Hybrid MT

As mentioned in previous sections, rule-based MT and statistical MT approaches have different advantages and disadvantages. The motivation behind the idea of hybridization is to complement the advantages of rule-based and statistical approaches. The rule-based approach tries to understand the basic principles of each language and encodes the principles and the transfer into rules. In the statistical approach, large quantities of parallel corpus is analyzed and a translation model is learned. The former approach claims that language is too complex to be represented by alignments and the latter approach claims that language is too complex to be represented by abstract rules.

Recent systems (Llitjos & Vogel, 2007; Oepen et al., 2007) try to combine linguistic and statistical techniques in a single system to improve translation quality. There are many challenges in developing a hybrid system: First, the resulting hybrid system should only inherit the best features of the two approaches rather than inheriting the worst features. Otherwise the translation quality will decrease. Next, the complexity of the final system should be at a maintainable level. Combining different technologies into a single system will introduce some complexity which should not obstruct the benefits. Finally, how the two ap-

proaches are combined is also very important, e.g. complementing a rule-based MT with stochastic tools versus complementing a statistical MT with structural tools. The degree of hybridity inside each component is likewise significant. For example, a transfer based MT is fully hybrid if at each step of the processing (i.e. analysis, transfer and generation), structural and statistical tools are used in combination. These challenging problems must be overcome so that the hybridization becomes feasible.

Figure 2.5 shows the relation between rule-based MT, statistical MT and hybrid MT. By definition, an MT system is said to be hybrid if it utilizes structural and statistical tools together. A system is described to be fully hybrid, only if structural and statistical tools are included in a balanced manner. As depicted in Figure 2.5, a hybrid MT can be located in a wide range in terms of the exploited technologies.

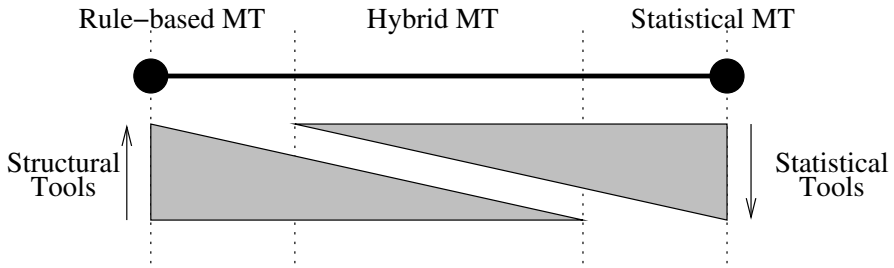

Figure 2.5: The relation between hybrid MT and others.

2.4 Speech Translation

Figure 1.1 represents the main goal of the speech translation (ST) task. In order to build an ST application, different components such as ASR and MT have to work in a composed manner. Depending on the type and tightness of such integration, the input to the MT component can be different from the text input. In many ST systems, including Mathias and Byrne (2006), Ney (1999), Vidal (1997), and Lavie et al. (1996), the MT is fed with the "speech input", which is actually a word graph containing many possible recognition hypotheses.

In state-of-the-art speech translation systems, the implementation is not just a simple sequential operation where speech is first transformed into text form and then translated into the target language. If implemented in this manner, then the probability of translating a text with recognition errors is high. A proper coupling mechanism should allow for the handling of possible recognition errors inside the translation component (Ney, 1999). In other words, the coupling mechanism should not restrict the speech translation system to one single recognized text; instead it should tolerate recognition errors to a certain extent. We explore in detail different coupling approaches used in ST in the following section.

2.4.1 Coupling

In Harper et al. (1994), the authors use the term "coupling" to define the integration of the language model with the speech recognition system. We use the same terminology to indicate the integration of the translation system with the speech

recognition system. As mentioned briefly in Section 1.1, Harper et al. (1994) classified the level of integration into three categories: tightly-coupled, semi-tightly coupled, and loosely-coupled. The representation in Figure 2.6, taken from Ringger (1995), depicts the architectural differences in the three approaches.

In a tightly-coupled system, the border between the recognizer and the parser is barely identifiable. The input to the system is processed incrementally at all levels. Tight coupling has a psycholinguistic motivation in the sense that humans seem to perform acoustic and linguistic analysis similarly in an incremental manner (Ringger, 1995). Knowledge-sources are shared by the recognizer and the parser. Tightly-coupled systems are hard to scale because of the inseparable and complex architecture. This method of coupling is not feasible for intricate language models because of the difficulty of implementation.

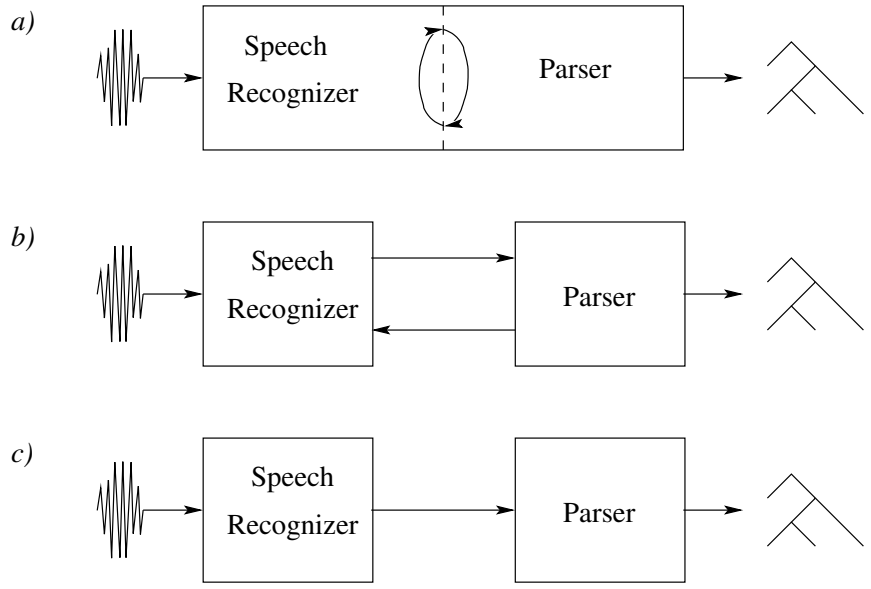

Figure 2.6: Tightness of coupling: a) tight, b) semi-tight, and c) loose.

Loosely-coupled systems are constructed from available components that were independently designed and implemented. This is done with a software engineering motivation. In such systems, components try to resolve ambiguities independently. Each module has its own isolated knowledge-source; these sources are not shared between the components. There is a one-way data flow from the recognizer to the parser. In this approach, it is considerably easier to scale the components because of this independence. Loosely-coupled systems are feasible for large problems and complex language models. Loose coupling enables the utilization of different approaches in each component. A statistically-based recognizer can be coupled with a rule-based parser in a loose manner.

Semi-tight coupling is somewhere between the other two approaches in terms of the degree of tightness, scalability and computational complexity. There is a two-way information exchange between the recognizer and the parser. Table 2.1 summarizes the features for all three coupling approaches.

2.5 Parsing

Parsing, in its simplest form, is the process of converting an input string to a structural representation. A more descriptive definition of parsing would be "the process of determining if a string of tokens can be generated by a grammar" (Aho et al., 1986). Before going into the details of parsing, we give some basic definitions of related key terms in parsing.

From the theory of computation perspective, a formal language is the set

Table 2.1: Summary of features for coupling approaches (Ringger, 1995).

FEATURE	TIGHT	SEMI-TIGHT	LOOSE
Motivation	Psycholinguistic evidence	Compromise	Software Engineering
Modularity of Knowledge-Sources	All KSs integrated in single model	KSs can be removed but not isolated	Isolated KSs
Inter-module Communication	N/A	Two-way	One-way
Scalability	Hard	Reasonable	Easy
Computation Complexity	Feasible only with simple language models	Feasible for models of moderate complexity	Feasible for large problems and complex language models

of strings over an alphabet (Σ). The set of acceptable forms define the syntax of the language (Sudkamp, 1991). Grammars are used to specify which forms are syntactically correct or incorrect. The sentences that can be derived by using a specific grammar are said to be grammatical sentences. Ungrammatical sentences cannot be derived according to that specific grammar. Similarly, a natural language can be regarded as the infinite set of sentences over the words of the language. A grammatical sentence in Turkish means that the sentence can be derived using the words and rules of the Turkish grammar.

For the purpose of parsing natural languages, we are interested in context-free

grammars (CFG). A CFG is defined using a 4-tuple $(\mathcal{V}, \Sigma, \mathcal{R}, \mathcal{S})$:

- $\mathcal{V}$ is the finite set of variables, or non-terminal elements.
- Σ is the finite set of terminal symbols disjoined from $\mathcal{V}$.
- $\mathcal{R}$ is the finite set of rules where each rule is a production of the form $\mathcal{A} \rightarrow a$. $\mathcal{A}$ is a non-terminal from $\mathcal{V}$ and a is a string of variables and terminals.
- $\mathcal{S}$ is the starting symbol from $\mathcal{V}$.

Table 2.2: Grammar G_1 describing a small fragment of Turkish.

1.	Sentence	→	Noun-Phrase Verb-Phrase
2.	Noun-Phrase	→	Pronoun
3.	Noun-Phrase	→	Noun
4.	Noun-Phrase	→	Pronoun Noun
5.	Verb-Phrase	→	Verb-Phrase Adv-Phrase
6.	Verb-Phrase	→	Noun-Phrase Verb
7.	Verb-Phrase	→	Verb
8.	Adv-Phrase	→	Adverb
9.	Pronoun	→	*o*
10.	Noun	→	*kitap*
11.	Adverb	→	*dün*
12.	Verb	→	*okudu*

The grammar G_1 in Table 2.2 is an example of a CFG describing a small fragment of the Turkish language. There are in total 12 rules or productions. The set of non-terminals $\mathcal{V}$ is {SENTENCE, NOUN-PHRASE, VERB-PHRASE, ADV-PHRASE, PRONOUN, NOUN, ADVERB, VERB}. The starting symbol of the grammar is SENTENCE. The set of terminals $\mathcal{S}$ is {*o, dün, kitap, okudu*}. Derivation is the process of substituting the grammar rules until no variable remains and the sentence is formed. Thus the derivation for the Turkish sentence *"o kitap okudu"* (*he read [a] book*, literally *"he book read"*), according to context-free grammar G_1 is as shown in Table 2.3.

Table 2.3: Sample derivation according to grammar G_1.

SENTENCE	$\Rightarrow$	NOUN-PHRASE	VERB-PHRASE		*(rule 1)*
	$\Rightarrow$	PRONOUN	VERB-PHRASE		*(rule 2)*
	$\Rightarrow$	*o*	VERB-PHRASE		*(rule 9)*
	$\Rightarrow$	*o*	NOUN-PHRASE	VERB	*(rule 6)*
	$\Rightarrow$	*o*	NOUN	VERB	*(rule 3)*
	$\Rightarrow$	*o*	*kitap*	VERB	*(rule 10)*
	$\Rightarrow$	*o*	*kitap*	*okudu*	*(rule 12)*

In the derivation shown in Table 2.3, the non-terminal in the leftmost position is substituted at each step. This type of derivation is called leftmost derivation. Alternatively, in a rightmost derivation, the variable in the rightmost position is substituted at each step. Another aspect of the above derivation is that the

most recent non-terminal is substituted until it is resolved into its terminal constituents. This type of derivation is called depth-first derivation. The opposite is breadth-first derivation in which all of the non-terminals in a step are expanded only one level before proceeding to the next level downwards.

A parse tree is a graphical way of representing the derivation sequence. Figure 2.7 depicts the parse tree for the derivation of the Turkish sentence *"o kitap okudu" (he read a book)*. The starting symbol of the grammar is the root node at the top in the parse tree for a grammatical sentence. Leaf nodes at the bottom are the terminals, or the words in the language. The nodes in between the root and the leaves are built out of the non-terminal variables, or the syntactic categories in the grammar. A parse tree is an appropriate representation to decide whether a sentence is grammatical or not. The structure of the tree gives information about the derivations applied on the input, but it does not indicate the direction (*leftmost* vs. *rightmost*) or the strategy (*depth-first* vs. *breath-first*).

Finally, we define parsing as the process of searching for the derivation sequence for an input string according to a grammar. At the end of parsing, we convert the input into a structural representation like a parse tree.

2.5.1 Ambiguity

It is usually the case that, for a certain input, there is more than one possible derivation sequence. Thus, there will be many corresponding parse trees instead of one. The possibility of multiple parses is called ambiguity and it is quite

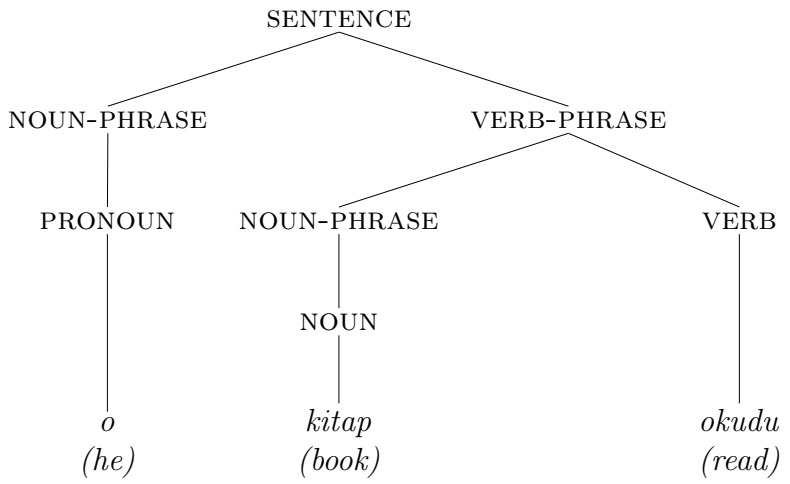

Figure 2.7: Parse tree for the derivation of *"o kitap okudu" (he read [a] book).*

common in natural language parsing. A different parsing indicates a different meaning. Syntactic knowledge is not enough to resolve ambiguities. Semantic clues and contextual information are also needed to deal with ambiguities.

Consider the same input *"o kitap okudu" (he read [a] book)* and the same grammar G_1. Another syntactically possible derivation is listed in Table 2.4. This new derivation requires one less substitution compared to the previous one. The English translation for the same input form with the below derivation would be *"that book read [sth]"*. Although this ambiguous derivation is grammatical according to G_1, it is meaningless in the sense that it states that an inanimate object reads something.

In general, a different derivation pattern is obtained by applying different rules. Thus, the decision to pick the appropriate rule becomes very important in

Table 2.4: Ambiguous derivation according to grammar G_1.

SENTENCE	⇒	NOUN-PHRASE		VERB-PHRASE	*(rule 1)*
	⇒	PRONOUN	NOUN	VERB-PHRASE	*(rule 4)*
	⇒	*o*	NOUN	VERB-PHRASE	*(rule 9)*
	⇒	*o*	*kitap*	VERB-PHRASE	*(rule 10)*
	⇒	*o*	*kitap*	VERB	*(rule 7)*
	⇒	*o*	*kitap*	*okudu*	*(rule 12)*

order to get the desired parsing. Figure 2.8 shows the alternative parse tree for the derivation listed above. Structural ambiguity can occur at any level during the parsing process. It is also possible that only one correct parse tree exists, but there are some local ambiguities that fail and do not reach the final parse tree. These kinds of local ambiguities introduce parsing inefficiency.

2.5.2 Top-down vs. Bottom-up Parsing

A top-down parser starts to build the parse tree from the topmost root node down to the leaves. The root node is expanded using the appropriate rules in grammar. The search for the final parse tree continues in this top-down manner until all the words in the input are assigned a leaf node. Parsing fails if a parse tree that spans all the words in the input sentence cannot be built; otherwise, parsing succeeds.

In bottom-up parsing, on the other hand, the building of the parse tree starts from the leaf nodes. Appropriate rules are applied, and the nodes are extended

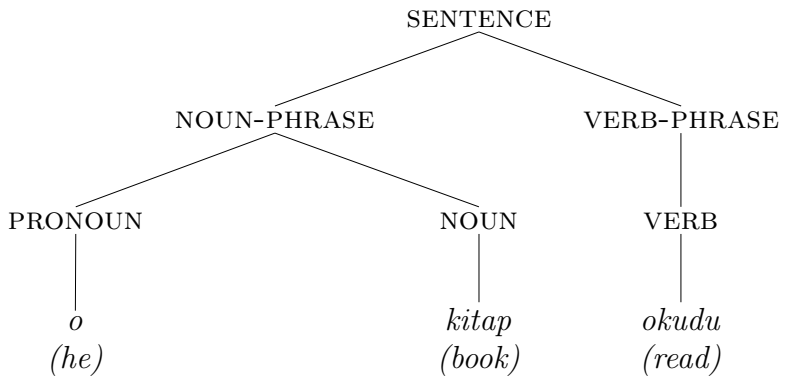

Figure 2.8: Parse tree for *"o kitap okudu" (that book read [sth])*.

from bottom to top toward the root node. Parsing succeeds if the root node, i.e. the starting variable $\mathcal{S}$ in grammar, can be reached and no words are left out of the tree.

In top-down parsing, there is the risk of building sub-trees that will not be part of the final parse tree. Still it does not waste time on sub-trees that do not lead to the starting variable $\mathcal{S}$. In bottom-up parsing, the opposite advantages and disadvantages exist. While it never builds sub-trees inconsistent with the input, this method wastes time on sub-trees that do not lead to $\mathcal{S}$. In order to avoid sacrificing the advantages of both approaches, the best parsing strategy would be to combine the two.

2.5.3 Chart Parsing

Because of the high cost of backtracking, algorithms with error recovery techniques are not utilized for parsing natural languages. Priority is given to algorithms employing dynamic programming; approaches such as the CKY parser (Younger, 1967), the Earley parser (Earley, 1970), or the chart parser (Kay, 1986). These parsers keep a table of partial solutions constructed and used during the analysis process.

Using a table or chart in parsing eliminates the need for backtracking by avoiding the multiplication effort. Another very important advantage of this technique is that it allows partial parsing of the input. This is a vital feature in syntactic parsing of natural languages in MT. Even if there is no successful parse tree for the whole sentence at the end of the analysis, different sub trees can be joined to cover the whole input. The use of a chart provides a compact representation for local ambiguities mentioned in Section 2.5.1.

The CKY parser is a bottom-up parser. It requires the grammar to be in Chomsky Normal Form (CNF), where each rule is in the form of $\mathcal{A} \rightarrow \mathcal{B}\ \mathcal{C}$ or $\mathcal{A} \rightarrow a$. This requirement forces the original grammar to be transformed into CNF before parsing. Thus, the resulting parse tree will include different categories that do not exist in the initial grammar. Additional processing is needed to convert back to a parse tree consistent with the grammar. The details of the CKY parser have been omitted, since it is out of the scope of this thesis.

A chart is a collection of nodes connected by edges labeled with syntactic

category information. Figure 2.9 represents a sample chart in a graphical way. As new edges are constructed, they are inserted into the chart. Duplicate edges are not allowed in the chart. Note that the chart contains edges from both the parse tree in Figure 2.7 and the parse tree in Figure 2.8. For the purpose of simplicity, some edges in the chart have been discarded.

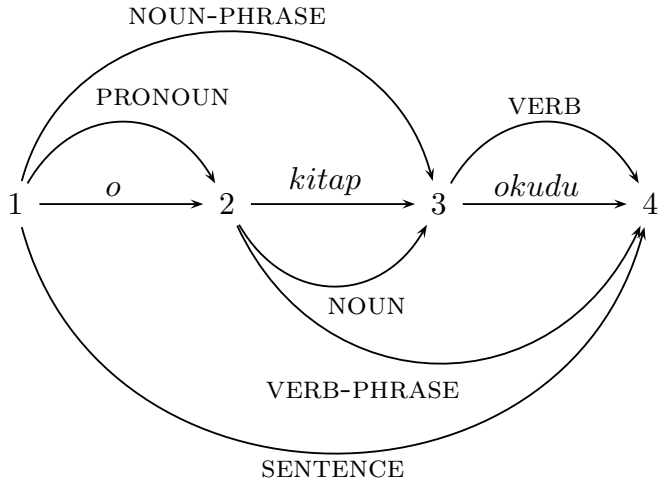

Figure 2.9: Graphical representation of a sample chart.

In an active chart parser, the parsing process is guided in a dynamic manner by an agenda which sets the order of the parsing. An Earley chart parser is passive in the sense that the processing is static. The parser proposed in (Kay, 1986) uses an agenda and is active because the processing is dynamic. In this thesis, whenever a chart parser is mentioned, an active chart parser should be understood, unless otherwise specified.

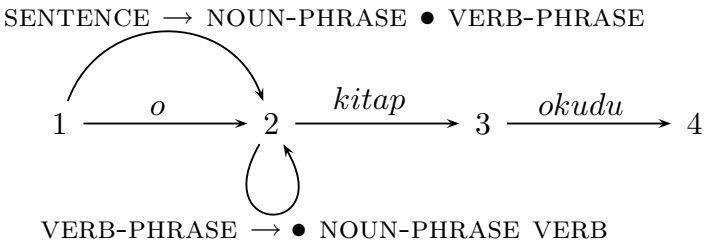

Figure 2.10: Chart containing incomplete edges.

In Figure 2.9, only complete edges are shown. However, a chart also contains incomplete or active edges. An incomplete edge represents a stage during processing and it is usually illustrated with a dotted rule. The variables on the left of the dot indicate that they are consumed at that specific stage. The variables on the right of the dot are the remainder parts, waiting to be matched. Figure 2.10 depicts a chart with two incomplete edges.

Algorithm 2.1 presents the chart parsing algorithm in pseudo-code format. The agenda and the chart are the two essential data structures in the parser. The agenda is simply a list of edges waiting to be inserted into the chart. Before any edge is inserted into the chart, it is put into the agenda. The strategy used in the agenda determines the order that the edges are added to the chart. Using a stack data structure in the agenda results in a depth-first search strategy. Likewise, a queue results in a breath-first search strategy.

The algorithm starts with the initialization of the chart and agenda. Addi-

Algorithm 2.1: The chart parsing algorithm.

```
input: grammar, sentence                    output: chart

algorithm Chart–Parse
    Initialize (chart, agenda, sentence)
    while agenda is not empty
        edge ← Pop (agenda)
        Process–Edge (edge)
    end while
end algorithm

procedure Process–Edge (edge)
    Push (chart, edge)
    Fundamental–Rule (edge)
    Predict (edge)
end procedure
```

tionally, the input words are looked up in the source lexicon, and for each word, a complete edge is inserted into the agenda. As a general rule, an edge is inserted into the chart and agenda, only if not done so before. Thus, it is guaranteed that any given edge is processed only once. After initialization, the **while** loop pops up one edge at a time from the agenda and processes it. The algorithm terminates if the agenda is empty.

At the first step of Process-Edge, the edge parameter is inserted into the chart. Next, the fundamental rule of chart parsing is applied to the current edge. This rule checks for two adjacent edges; the first of which should be incomplete and the second of which should be complete. If this condition is satisfied, and if

the first variable in the remainder of the incomplete edge and the rule variable of the complete edge match, then a new edge spanning both edges is inserted into the agenda. The dot in the incomplete edge is advanced to the appropriate position in the new edge.

After the execution of the FUNDAMENTAL-RULE, listed in Algorithm 2.2, new edges, based on the current edge, are predicted. The prediction is the creation of new edges to be inserted into the agenda according to a strategy. Top-down, bottom-up, or an integrated strategy can be followed. Finally, after the algorithm terminates and the chart is filled, the parse tree is extracted. The extraction begins with the starting symbol, and all constituent edges are retrieved by tracing down until the leaf levels are reached.

In Stock et al. (1988), the concept of bidirectional charts is introduced. Instead of starting from the beginning of the sentence and extending toward the end of the sentence, the processing starts from previously determined words, called "islands", and extends in both directions. This enables the processing to be carried out in a selective manner.

2.5.4 Unification Based Parsing

The basic form of CFG is satisfactory for modeling formal languages, but it is rarely adequate for natural languages. In order to represent complex phenomena existing in natural languages, some extensions are required. Most of the time, it is insufficient to use only the generalized forms of syntactic categories in grammar.

Algorithm 2.2: Fundamental rule of chart parsing and the prediction procedure.

```
procedure FUNDAMENTAL–RULE (A → α • D, [j,k])
    if D = Bβ // edge is incomplete
        for each (B → γ•, [k,l]) in chart
            PUSH (agenda, (A → αB • β, [j,l]) )
        end for
    else          // edge is complete; D is empty
        for each (C → γ • Aβ, [i,j]) in chart
            PUSH (agenda, (C → γA • β, [i,k]) )
        end for
    end if
end procedure
procedure PREDICT (A → α • D, [j,k])
    if D is null   // edge is complete
        for each C → Aβ in grammar
            PUSH (agenda, (C → A • β, [j,k]) )
        end for
    else IF D = Bβ // edge is incomplete
        for each B → γ in grammar
            PUSH (agenda, (B → •γ, [k,k]) )
        end for
    end if
end procedure
```

Additional features in the constituents are required to decide whether a sentence is grammatical or not. For example, in Turkish, the subject and the verb in a phrase have to agree with each other in terms of the PERSON and NUMBER categories. A possible solution to formalize this requirement in the grammar is to introduce new variables which reflect the new information (e.g. NOUN-1-SG and VERB-1-SG). Then, to account for the subject-verb agreement, we can write

a new rule as shown below:

$$\text{VERB-PHRASE} \rightarrow \text{NOUN-1-SG VERB-1-SG} \tag{2.3}$$

However, this approach is not promising, considering the excessive number of rules and variables that need to be inserted into the grammar. A better approach would be to associate each variable with a list of features and values. These feature-value lists are called the attribute-value matrix (AVM). The values associated with each attribute in the AVM can be simple (e.g. atomic string, id) or complex (e.g. list). This extension with features and values leads to an augmented grammar. The rule in (2.3) can be rewritten as follows:

$$\text{VERB-PHRASE} \rightarrow \begin{matrix}\text{NOUN}\\ \begin{bmatrix}\text{PERSON} & 1\\ \text{NUMBER} & sg\end{bmatrix}\end{matrix} \quad \begin{matrix}\text{VERB}\\ \begin{bmatrix}\text{PERSON} & 1\\ \text{NUMBER} & sg\end{bmatrix}\end{matrix} \tag{2.4}$$

It is not enough to associate variables in the grammar with feature-value lists. Additionally, there is a need for a mechanism to make use of these features. Therefore, parsers are extended with functional power in order to employ the associated AVMs. This functionality is used to formulate constraints on the rules. In this augmented parsing approach, for the problem of subject-verb agreement, a constraint can be specified so that the rule will succeed only if the associated AVMs unify successfully. This unification mechanism is also used for other purposes. To make use of the feature-values in subsequent rules, the features are propagated toward newly built categories. The AVM structures coming from

the constituents are unified into one AVM, and that newly built structure is associated with the rule variable.

$$\begin{bmatrix} \text{FORM} & \textit{'okudu'} \\ \text{TENSE} & \textit{past} \\ \text{NUMBER} & \textit{sg} \\ \text{PERSON} & \textit{1} \\ \text{SUBJ} & \begin{bmatrix} \text{FORM} & \textit{'o'} \\ \text{NUMBER} & \textit{sg} \\ \text{PERSON} & \textit{1} \end{bmatrix} \\ \text{OBJ} & \begin{bmatrix} \text{FORM} & \textit{'kitap'} \\ \text{NUMBER} & \textit{sg} \\ \text{ANIMATE} & \textit{minus} \end{bmatrix} \end{bmatrix}$$

Figure 2.11: Attribute value matrix.

These extensions give the parser a Turing-equivalent power, which is more than enough to parse any context-sensitive grammar, where natural languages are placed in the Chomsky-hierarchy. It has to be noted that parsing with unification-based grammars is more expensive compared to standard parsing approaches. Additional constraint operations introduce an extra processing load on the parser; however, the value they add is worth the price.

2.5.5 Optimization in Parsing

The number of rules required in a grammar to model a natural language can range from a couple hundred (if using unification-based parsing) to more than 10.000 (if using direct parsing of CFGs). In any case, efficiency is a significant issue in parsing. As a fundamental principle, any possible analysis is inserted into the chart, whether it is useful at the end of parsing or not. The aim in optimization is to get rid of the edges that can be predicted to be ineffectual.

Bottom-up filtering, a strategy used in top-down parsing, aims to optimize the parsing process. The strategy is introduced in Rosenkrantz and Lewis (1970) and it is based on the detection of the first constituent of a phrase which is the left-corner of the tree. A table containing the left-corner categories for each variable is constructed before parsing. This table is consulted during parsing to block edges that will fail in the long term. In a more formal way, no edge of the form $(\mathcal{A} \rightarrow \alpha \bullet \mathcal{B}, [i, j])$ is inserted into the agenda unless α_{j+1} is a left-corner of B. The implementation is done by adding new constraints to the fundamental rule and to the prediction process of chart-parsing. The new constraints check left-corners before inserting any edge into the agenda. Blache and Morin (1990) propose a parser which combines bottom-up filtering and unification-based parsing.

In Moore (2004), a chart parser for non-augmented CFG grammars is presented. Different additional optimization strategies are combined in this parser. The author claims to achieve improvements that result in increases in speed averaging 38% or more.

CHAPTER 3

THE SPEECH TRANSLATION SYSTEM

The general architecture of the proposed speech translation system is depicted in Figure 3.1. The system is loosely coupled; in other words, there is a one-directional information flow between the SR and MT. The original word graph created by the SR is determinized and minimized according to FSM minimization algorithms (Mohri & Riley, 1997) and N-best hypotheses are extracted based on acoustic scores (Mohri et al., 1998). The extracted data format is again a word graph. The pruned word graph is processed by the MT component (described in Her (1996)) of the speech translation system.

The MT task deploys a transfer-based approach, and processing is divided into three clear-cut phases: analysis, transfer and generation. At the end of the analysis, any ambiguity is resolved and the best-sentence hypothesis is chosen for the transfer stage. The analysis is accomplished in two consecutive tasks. First, morphological analysis is performed at the word level, and any information carried by the word is extracted to be used in later stages. Next, syntactic analysis is performed at the sentence level.

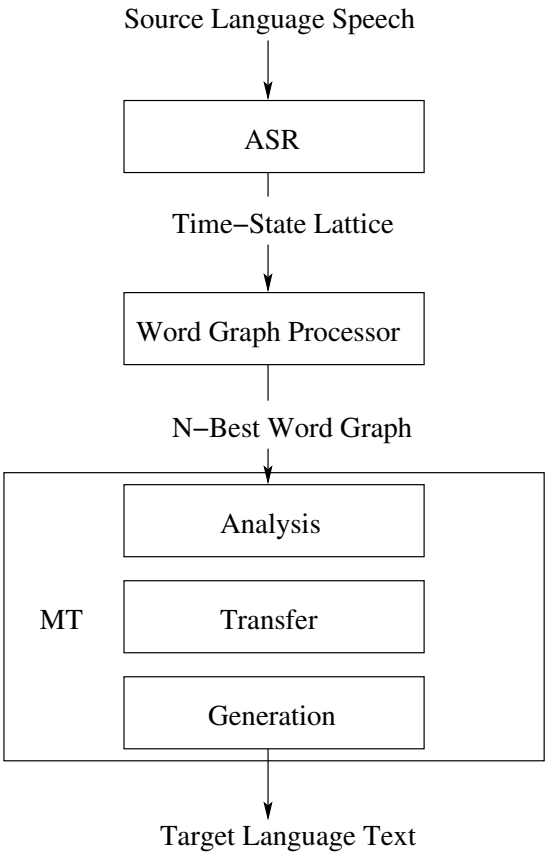

Figure 3.1: Speech Translator.

3.1 The ASR and the Word Graph Processor

The ASR component used in the proposed architecture can be any ASR system that outputs hypotheses in word graph format. All state-of-the-art ASR systems produce word graph format. Thus, the speech translation architecture is not bound to a specific ASR system. Details of the word graph processing component is explored in Chapter 4.

3.2 Morphological Analysis

A chart parser augmented with unification-based feature structures (FS) lies in the heart of the morphological analyzer. The parser is responsible for both analysis and generation tasks. Morphological rules are used to derive the uninflected form of an inflected word during analysis. An imaginary rule definition is given in Algorithm 3.1. The sample rule states that a complete edge with category 'WORD-CAT' is built if all essential elements in the definition are matched. A morpheme label is defined with a FS and a set of string operations. A morpheme is matched if at least one of the string operations in the set succeeds. Moreover, the FSs of all matching morphemes and the FS that comes from the lexical stem should unify. A morpheme with a non-unifying FS is not matched.

Algorithm 3.1: Imaginary morphology rule definition.

```
MORPH–RULE ::
    STEM
    ( MORPH1 )
    { MORPH2 MORPH3 }
    MORPH4 *
    < MORPH5 MORPH6 >
    →
    WORD–CAT
```

The rule definition is composed of morpheme categories bundled with regular expression formalism. Parentheses represent optionality; the asterisk and the plus are the signs for *kleene*-star and *kleene*-plus, respectively. Disjunction is rep-

resented by curly brackets and morpheme categories listed inside angle brackets represent free ordering. The regular expression formalism is recursive; disjunctive or optional elements can also contain disjunctive or optional elements. A valid ordering that can be matched with the rule in Algorithm 3.1 could be 'MORPH5 MORPH6 MORPH2 STEM'. It is important to note that the ordering does not imply concatenative morphotactics; instead, the ordering refers to the sequence of the application of the morpheme operations, which might or might not be concatenative, on the word.

Edges in the chart represent arcs spanning morpheme categories and the lexical category of a word. A complete edge remaining inactive in the chart describes a successfully matched word with its lexical stem and all its morphemes. An incomplete edge represents already matched morphemes and is active in the chart and is looking for other morphemes or a lexical stem in order to build a complete edge.

Morphological rules in the grammar are compiled into deterministic FSMs. Each morpheme category in the rule definition corresponds to an arc in the FSM. During analysis, processing starts from the last morpheme category in the rule definition and continues toward the stem. The morphemes used in the rule definition are associated with allomorph tables and typed feature structures. Each row in the allomorph table contains an allomorph described by a set of morphological string operations.

The successful matching of a morpheme indicates that at least one of the string operations in the corresponding allomorph table is performed successfully.

String operators are unary functions that allow prefixal, suffixal, circumfixal and infixal modification of the word. String modifications applied during analysis and generation employ opposing operations (e.g., *cut* vs. *add*, *geminate* vs. *degeminate*). In analysis, feature structures associated with the matching arc are unified in order to build the final feature structure. A similar system integrating feature structures and a finite-state model is presented in Zajac (1998).

3.3 Syntactic Parsing

The syntactic analyzer consists of a chart parser in which the rules modeling the source language grammar are augmented with feature structures. The grammar is implemented using *Lexical Functional Grammar* (LFG) paradigm. The primary data structure to represent the features and values is a *directed acyclic graph* (dag). The system also includes an expressive Boolean formalism, used to represent functional equations to access, inspect, or modify features or feature sets in the dag. Complex feature structures (e.g., lists, sets, strings, and conglomerate lists) can be associated with lexical entries and grammatical categories using inheritance operations. Unification is used as the fundamental mechanism to integrate information from lexical entries into larger grammatical constituents.

3.3.1 Bidirectional Chart Parsing

The chart parsing algorithm given in Algorithm 2.1 processes input texts from left to right. Given a grammar rule like $\mathcal{A} \rightarrow \mathcal{BCD}$, the parser tries to match

first $\mathcal{B}$ then $\mathcal{C}$ and finally $\mathcal{D}$. The beginning state is expressed with a dotted rule as $\mathcal{A} \rightarrow \bullet\mathcal{BCD}$. The dot is advanced one position at a time while each category is matched. Different from this approach, in Steel and de Roeck (1987), the idea of bidirectional parsing is introduced. In bidirectional parsing, processing can start on any selected category in the right-hand-side of the grammar rule. Then, it continues outwards from the selected category, called *trigger*, to both directions. The states in bidirectional parsing are represented with double-dotted rules. Assuming that $\mathcal{C}$ is the *trigger*, the state in which the *trigger* is matched is represented as $\mathcal{A} \rightarrow \mathcal{B} \bullet \mathcal{C} \bullet \mathcal{D}$. At a next state, the first dot is advanced to the left if $\mathcal{B}$ is matched or the second dot is advanced to the right if $\mathcal{D}$ is matched.

Bidirectional parsing requires modifications on the original chart parsing algorithm. Algorithm 3.2 shows the updated FUNDAMENTAL-RULE procedure in bidirectional chart parsing. Basically, it checks adjacent edges on both dots. Thus, the processing continues to both directions after starting from the trigger category.

Similarly, the PREDICT procedure in the one directional version of the chart parsing algorithm should be adapted to work in a bidirectional way. Algorithm 3.3 shows the updated BD-PREDICT procedure. If the parameter to the procedure is a complete edge and if the category built by that complete edge matches the trigger in a bottom-up rule, then a new edge is created and inserted into the agenda with that bottom-up rule. However, if the parameter is an incomplete edge, then the top-down rules that build the categories expected in both directions are inserted into the agenda as new edges.

Algorithm 3.2: Updated fundamental rule in bidirectional chart parsing.

```
procedure BD–Fundamental–Rule (A → B • α • C, [j, k])
    if B = βD          // edge is incomplete
        for each (D → •δ•, [i, j]) in chart
            Push (agenda, (A → β • Dα • C, [i, k]) )
        end for
    end if
    if C = Dγ          // edge is incomplete
        for each (D → •δ•, [k, l]) in chart
            Push (agenda, (A → B • αD•γ, [j, l]) )
        end for
    end if
    if B is null and C is null // edge is complete
        for each (D → βA • γ • δ, [k, l]) in chart
            Push (agenda, (D → β • Aγ • δ, [j, l]) )
        end for
        for each (D → β • γ • Aδ, [i, j]) in chart
            Push (agenda, (D → β • γA • δ, [i, k]) )
        end for
    end if
end procedure
```

3.3.2 Constituent Structure and Functional Structure

The constituent structure (c-structure) represents the composition of syntactic constituents for a phrase. It is the term used for parse tree in LFG. The functional structure (f-structure) is the representation of grammatical functions in LFG. Attribute-value-matrices are used to describe f-structures. A sample c-structure and the corresponding f-structures in English are shown in Figure 3.2.

Algorithm 3.3: Updated prediction procedure in bidirectional chart parsing.

```
procedure BD–PREDICT (A → B • α • C, [j, k])
    if B is null and C is null // edge is complete
        for each D → βAγ in grammar where A is trigger
            PUSH (agenda, (D → β • A • γ, [j, k]) )
        end for
    else
        if B = βD // edge is incomplete
            for each D → γ in grammar
                PUSH (agenda, (D → γ•, [j, j]) )
            end for
        end if
        if C = Dγ // edge is incomplete
            for each D → γ in grammar
                PUSH (agenda, (D → •γ, [k, k]) )
            end for
        end if
    end if
end procedure
```

For simplicity, many details and feature values are not given. The dag containing the information originated from the lexicon and the information extracted from morphological analysis is shown on the leaf levels of the parse tree in Figure 3.2. The final dag corresponding to the root node is built during the parsing process in cascaded unification operations specified in the grammar rules.

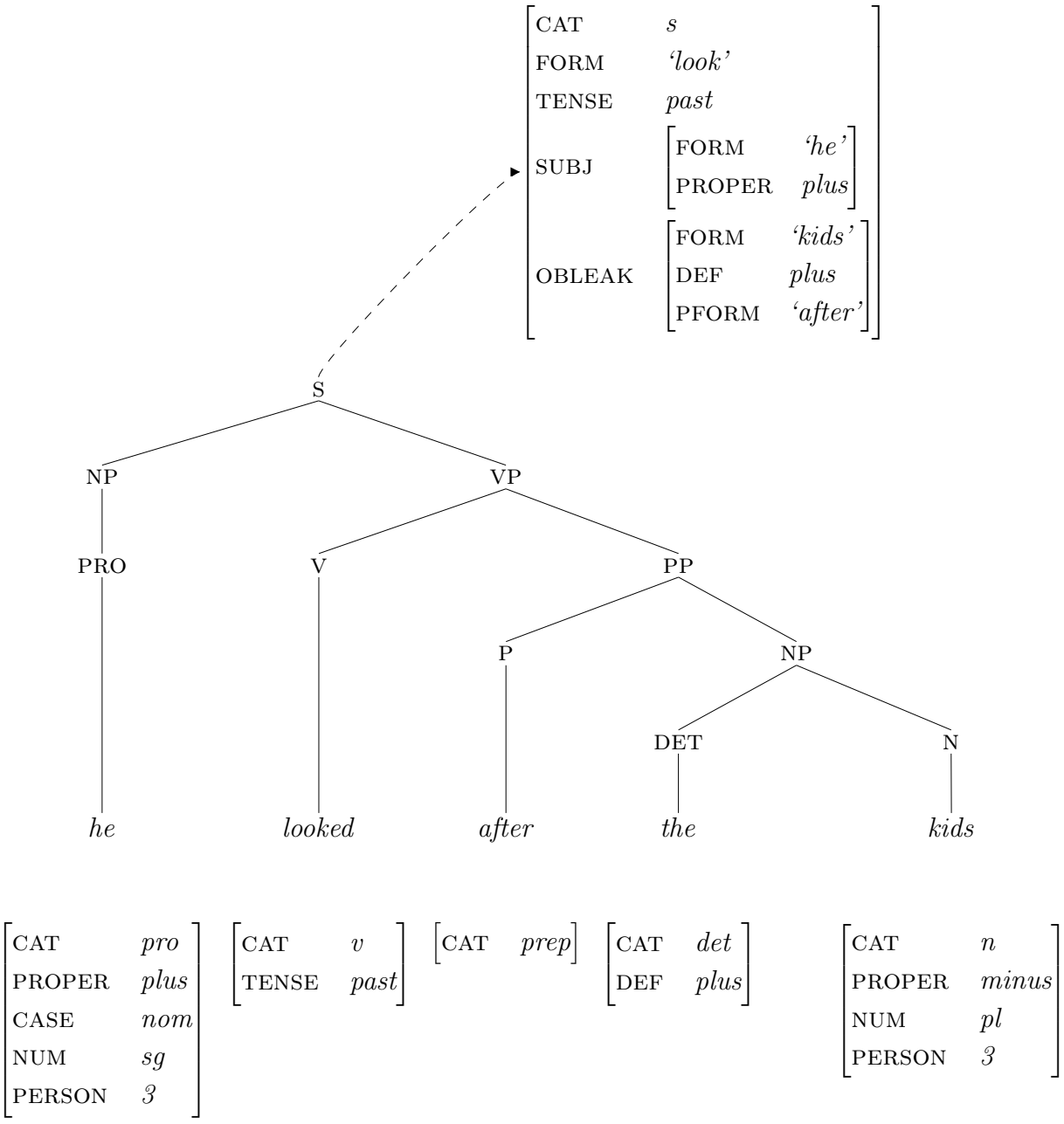

Figure 3.2: The c-structure and the associated f-structures.

3.3.3 Parse Evaluation

After all rules are executed and no more edges are left in the agenda, the chart parsing process ends and parse evaluation begins. The chart is searched for com-

plete edges with the final symbol of the grammar (e.g. SBAR) as their category. Any such edge spanning the entire input represents the full parse. If there is no such edge then the parse recovery (Section 3.3.4) process takes control.

If the input sentence is ambiguous, then, at the end of parsing, there will be multiple parse trees in the chart that span the entire input. Similarly, a grammar built with insufficient constraints can lead to multiple parse trees. In this case, all possible edges are evaluated for *completeness* and *coherence* (Bresnan, 1982) starting from the edge with the highest weight. A parse tree is *complete* if all the functional roles (SUBJ, OBJ, SCOMP etc.) governed by the verb are actually present in the c-structure; it is *coherent* if all the functional roles present are actually governed by the verb. The parse tree that is evaluated as *complete* and *coherent* and has the highest weight is selected for further processing.

3.3.4 Parse Recovery

In general, a parsing process is said to be successful if a parse tree can be built according to the input sentence. The building of the parse tree fails when the sentence is ungrammatical. For the goal of MT, however, a parse tree is required for the transfer stage and the generation stage even if the input is not grammatical. Therefore, for any input sentence, a corresponding parse tree is built at the end of parsing.

If parsing fails, i.e. if all rules are exhausted and no successful parse tree has been produced, then the system tries to recover from the failure by creating a

tree like structure. Appropriate complete edges in the chart are used for this purpose. The idea is to piece together all partial parses for the input sentence, so that the number of constituent edges is minimum and the weight of the final tree is maximum. While selecting the constituents, overlapping edges are not chosen.

The recovery process functions as follows:

- The whole chart is traversed and a complete edge is inserted into a candidate list if it has the highest weight for that start-end position. If two edges have the same weight, then the farthest one to the leaf level is preferred.
- The candidate list is traversed and a combination with the minimum number of constituents is selected. The edges with the widest span get into the winning combination.
- The c-structures and f-structures of the edges in the winning combination are joined into a whole c-structure and f-structure which represent the final parse tree for the input.

3.4 Transfer

Transfer is the process of converting source language representations (i.e. c-structure and f-structure) into target language representations. The parse tree, either selected at the end of parse evaluation or built at the end of parse recovery, is further processed to be used in generation. Processing is directed by transfer rules contained in transfer entries. The differences between the functional

structure of the source language and that of the target language are resolved at the end of the transfer. Figure 3.3 depicts a sample mapping from English to Turkish f-structures. There are two types of dissimilarities between the source and target f-structures. First, English word forms are replaced (Section 3.4.1) with Turkish word forms. Second, f-structure created according to the English language analysis rules is modified (Section 3.4.2) so that it complies with proper Turkish syntax analysis.

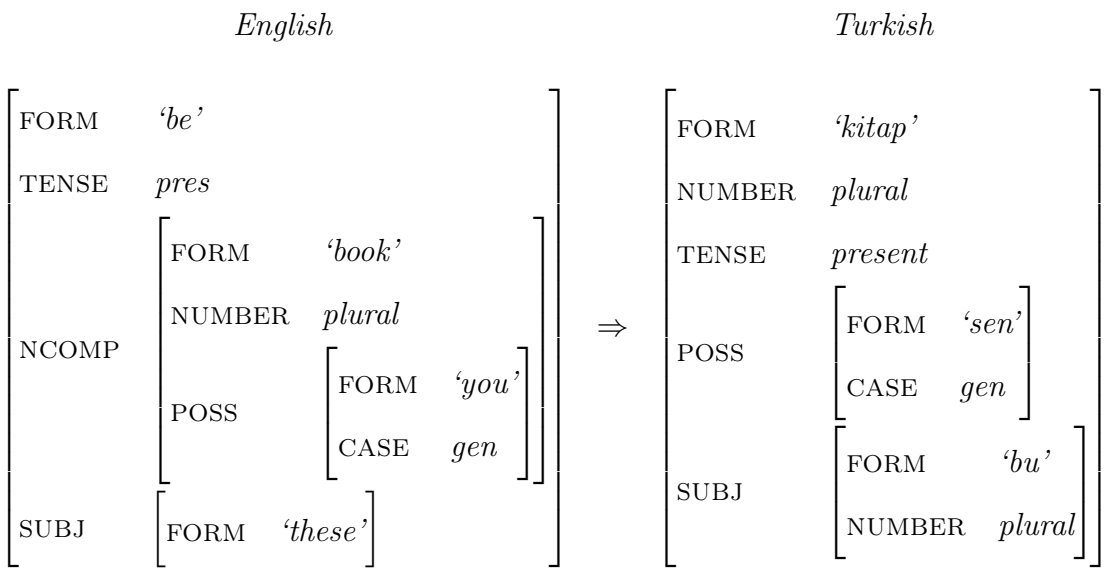

Figure 3.3: Source language to target language transfer.

3.4.1 Word replacement

Source language words represented in the f-structure under the label of FORM are looked up in the transfer lexicon. If the word has multiple senses in the lexicon, then the transfer meaning corresponding to the same category as in the analysis is selected. The source language meaning is replaced with the selected target language meaning. Figure 3.4 shows an abstract representation of the English-Turkish transfer entry *'book'*. The word *'book'* is analyzed as in the sentence *'these are your books'* as shown in Figure 3.3. Thus, one of the transfer meanings listed under the category NOUN in the transfer entry is chosen and inserted into the dag. The selection among multiple meanings in a category is determined according to the selection rules contained in each meaning dag. Selecting the correct word translation is important as this affects the translation quality directly. So, selection rules must represent syntactic and semantic criteria properly.

$$
\textit{book}\left[\begin{array}{ll}
\text{NOUN} & \left[\begin{array}{ll}
\text{FE} & \textit{action-1} \\
\text{VALUE} & \left\{\left[\begin{array}{ll}\text{FORM} & \textit{'kitap'}\\ \text{SRULE} & \textit{selection-1}\end{array}\right], \left[\begin{array}{ll}\text{FORM} & \textit{'defter'}\\ \text{SRULE} & \textit{selection-2}\end{array}\right], \cdots\right\}
\end{array}\right] \\
\text{VERB} & \left[\begin{array}{ll}
\text{FE} & \textit{action-2} \\
\text{VALUE} & \left\{\left[\begin{array}{ll}\text{FORM} & \textit{'yer ayır'}\end{array}\right]\right\}
\end{array}\right]
\end{array}\right]
$$

Figure 3.4: Abstract representation of *'book'* transfer entry.

3.4.2 Structural transfer

The dag in Figure 3.3 undergoes structural modifications other than word replacement in order to be utilized in the generation module. In the English analysis of the sentence, the copula verb *'be'* is supplemented with a nominal complement at the head position in the dag. Copulative constructions in Turkish are formed without a verb, therefore, the head inside NCOMP in the analysis dag becomes the head of the entire dag. All the contents of NCOMP are moved one level up and the final dag is formed. The actions that modify the dag are specified in the transfer entry under label FE (functional expression). The selection rule can also contain dag manipulation expressions.

Any kind of deviation between the two languages should be resolved during this stage using the tools explained above. For example, if the source language does not mark GENDER information while the target language does, then default GENDER information should be inserted at this stage.

3.5 Text Generation

Any processing done prior to this step can be regarded as a preparation to generation. The aim in generation is to produce a sentence in the target language with proper word forms and in proper sequence. Three different jobs are performed in consecutive order to accomplish the generation task.

3.5.1 Target Word Lookup

FORM values in the f-structure are looked up in the target language lexicon and the contents are extended into the dag. This operation is necessary to get information specific to the target language generation task. For example, in Arabic, there exist around 300 different plural inflection paradigms for nouns. The target lexicon is referred to in order to decide on the inflection paradigm of a specific noun. Without the target lexicon look-up, it is impossible to determine the paradigm as this information does not exist in the f-structure coming from the analysis and transfer.

It is important, however, that any new information that would generate a sentence with a different meaning from the source sentence not be introduced into the dag. For example, if a source verb is not marked for tense, then no tense related information that can change the morphological generation of the target word should be augmented from the target lexicon.

3.5.2 Morphological Generation

The aim in morphological generation is to produce the inflected form of a word according to the features and values in the FS. The entire f-structure is traversed and all FORM values, which indicate the stem forms, are supplied to the associated morphology rules. At the end of the execution of the morphological rule, the inflected form of the word is inserted into the dag as a new feature-value. This label is used later to generate the target sentence. The Turkish f-structure in

Figure 3.3 is presented after morphological generation in Figure 3.5.

$$\begin{bmatrix}
\text{FORM} & \textit{'kitap'} \\
\text{IFORM} & \textit{'kitapların'} \\
\text{NUMBER} & \textit{plural} \\
\text{TENSE} & \textit{present} \\
\text{POSS} & \begin{bmatrix} \text{FORM} & \textit{'sen'} \\ \text{IFORM} & \textit{'senin'} \\ \text{CASE} & \textit{gen} \end{bmatrix} \\
\text{SUBJ} & \begin{bmatrix} \text{FORM} & \textit{'bu'} \\ \text{IFORM} & \textit{'bunlar'} \\ \text{NUMBER} & \textit{plural} \end{bmatrix}
\end{bmatrix}$$

Figure 3.5: Sample f-structure containing inflected forms.

From a practical point of view, morphological generation is the reverse action of morphological analysis. A rule similar to the analysis rule can be used to generate the desired word form. An important difference is in the direction of the elements' order of execution in the rule definition (explained in Section 3.2). This is achieved by reversing the corresponding FSM. The starting state and final states are switched and the directions of the arcs are changed.

Another deviation from analysis is in the string manipulation operations; reverse actions have to be carried out in generation. For example, *cut* operation in analysis is replaced with *add* operation and vice versa. In general, morphological generation can be described as a less difficult task compared to morphological

analysis. The search space in generation is greatly reduced because of the linguistic features available. The morphology rule and the morphotactic transformation to be applied can be deduced from the available FS. In contrast, in morphological analysis, all rules and transformations in the grammar have to be tested to determine whether they succeed or not.

3.5.3 Word Ordering

In the word ordering stage, parts of the f-structure are projected onto a target language sentence in accordance with ordering rules. The word ordering operation can be regarded as the reverse of parsing. A target language word ordering grammar similar to the one in analysis is used in generation. There are even some studies, e.g. Shieber (1988), that explore the usage of exactly the same grammar in both analysis and generation.

The idea in word ordering is to traverse the functional structure and to generate the strings in a head-driven fashion. The generation rules actually describe the order of the traversal. Similar approaches are presented in Shieber et al. (1989), Kay (1996) and Neumann (1998). The sentence is generated by applying the ordering rules at each level of the dag. A sample word ordering grammar is represented in Table 3.1. Note that the rules contain functional roles instead of the lexical categories.

Assuming that the rules in Table 3.1 are given, the derivation for the sample f-structure will be as shown in Table 3.2. Each line represents both the rule

Table 3.1: A small fragment of word ordering rules for Turkish.

#	LHS		RHS
1.	SENTENCE	→	COPULA-SENTENCE
2.	COPULA-SENTENCE	→	SUBJ POSS IFORM
3.	SUBJ	→	IFORM
4.	POSS	→	IFORM

that is applied and the derivation that is obtained at the end of the step. The process starts with the final symbol of the grammar. All non-terminal symbols are expanded with appropriate rules. The operation continues until the derivation string consists of all terminal symbols.

Table 3.2: Sample derivation.

#	Derivation String			Rule
1.	SENTENCE			
2.	COPULA-SENTENCE			*rule 1*
3.	SUBJ	POSS	IFORM$_{fs}$	*rule 2*
4.	IFORM$_{subj}$	POSS	IFORM$_{fs}$	*rule 3*
5.	*bunlar*	POSS	IFORM$_{fs}$	
6.	*bunlar*	IFORM$_{poss}$	IFORM$_{fs}$	*rule 4*
7.	*bunlar*	*senin*	IFORM$_{fs}$	
8.	*bunlar*	*senin*	*kitapların*	

CHAPTER 4

WORD GRAPH PROCESSING

In this chapter, the stages in word graph processing are explained. The elucidated stages start from the moment the ASR produces an output and continue until the final hypothesis is chosen at the end of the parsing process. The word graph plays a fundamental role in the proposed system in order to integrate statistical ASR with rule-based MT systems.

4.1 ASR Output

Speech recognition components produce output in a variety of different forms. The structure of the output depends on the intended use of the ASR component. If no further processing is required after ASR, then a single stream of recognized words can simply be generated. However, if there is a need for additional processing after ASR (e.g. in ST), then the generated output contains multiple hypotheses. N-best list is an ordinary way of representing multiple hypotheses in string form. Alternatively, a time-state lattice contains a lot more information than the N-best list. A sample time-state lattice is shown in Figure 4.1.

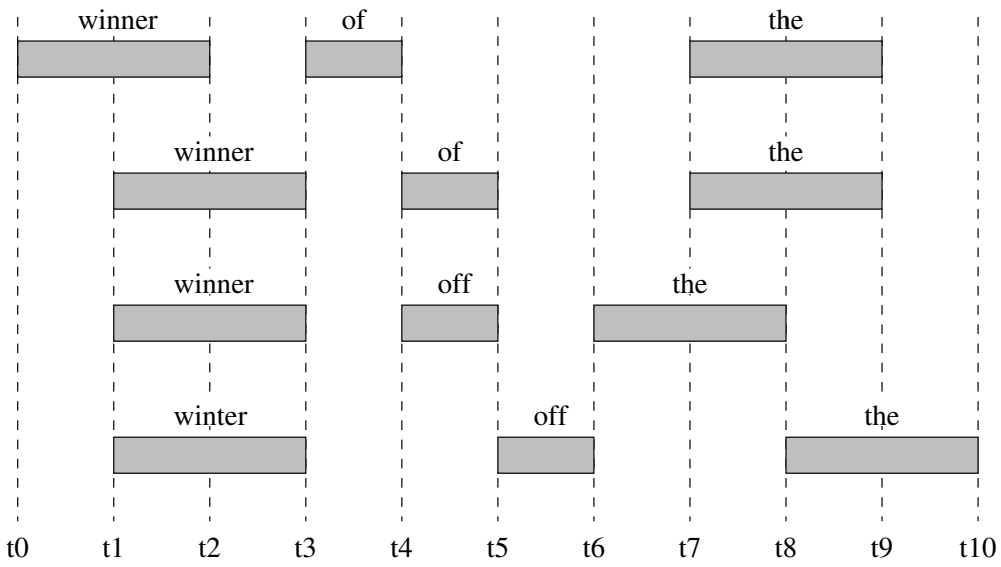

Figure 4.1: A sample time-state lattice.

4.2 Pruning the Word Graph

The word graphs produced by an ASR are far bigger than the one shown in Figure 4.1. A small-sized lattice from the NIST HUB-1 (Pallett et al., 1994) data set can easily contain a couple of hundred states and more than one thousand arcs. The smallest lattice $\mathcal{F}_1$ in the set is shown in Figure 4.2. Another lattice with 422 states and 1030 arcs, which is still far smaller than the largest one, from the same data set is depicted in Figure 4.3. The largest word graph in the NIST HUB-1 data set has 25 000 states and almost 1 million arcs. No unification-based chart parser is capable of coping with an input of this size. It is impractical and unreasonable to parse the FSM in the same form as it is output from the ASR. Instead, the word graph is pruned to a reasonable size so that it can be parsed according to acceptable time and memory limitations.

The pruning process contains the below listed steps in the specified order. The

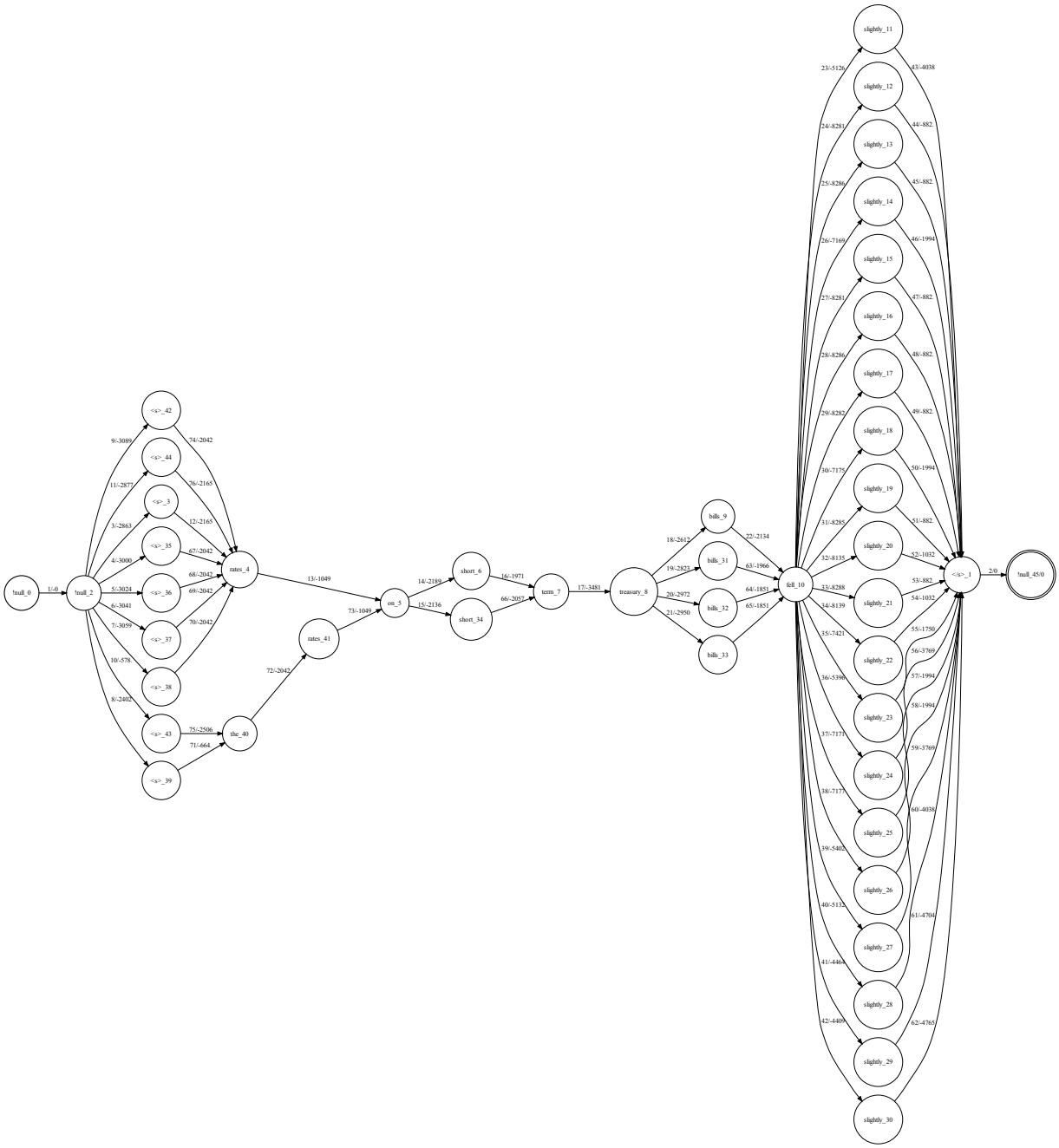

Figure 4.2: The smallest lattice $\mathcal{F}_1$ in the NIST HUB-1 data set.

tools used in stages 2 to 4 are taken from the 'AT&T FSM Library$^{\text{TM}}$' (Mohri et al., 1998).

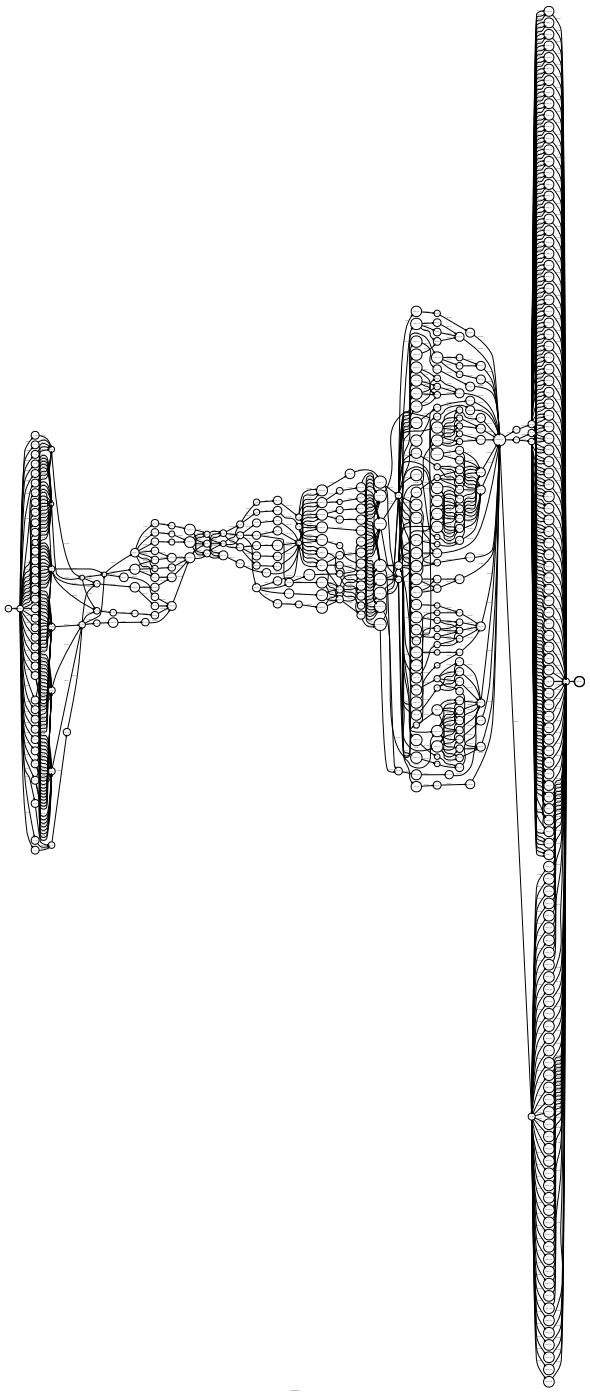

Figure 4.3: A sample mid-sized lattice from the NIST HUB-1 data set.

1. Word graph to FSM conversion.

2. FSM determinization

3. FSM minimization

4. N-best list extraction

The above steps are explained in detail in the coming sub-sections.

4.2.1 Word Graph to FSM Conversion

The pruning process starts by converting the time-state lattice to a finite state machine. This way, algorithms and data structures for FSMs are utilized in the following processing steps. Each word in the time-state lattice corresponds to a state node in the new FSM. The time slot information is also dropped in the recently built automata. The links between the words in the lattice are mapped as the FSM arcs.

In the original representation in a lattice, the word labels in the time-state lattices are on the nodes, and the acoustic scores and the statistical language model scores are on the arcs. Similarly, as depicted in Figure 4.2, the words are also on the nodes. This representation does not fit into the chart definition where the words are on the arcs. Therefore, the FSM is converted to an arc labeled FSM, $\mathcal{F}_2$, as shown in Figure 4.4. The conversion is accomplished by moving back the word label on a state to the incoming arcs. The start state in $\mathcal{F}_1$ is also trimmed. The weights on the arcs represent the negative logarithms

of probabilities. In order to find the weight of a path in the FSM, all weights on the arcs existing on that path are added up.

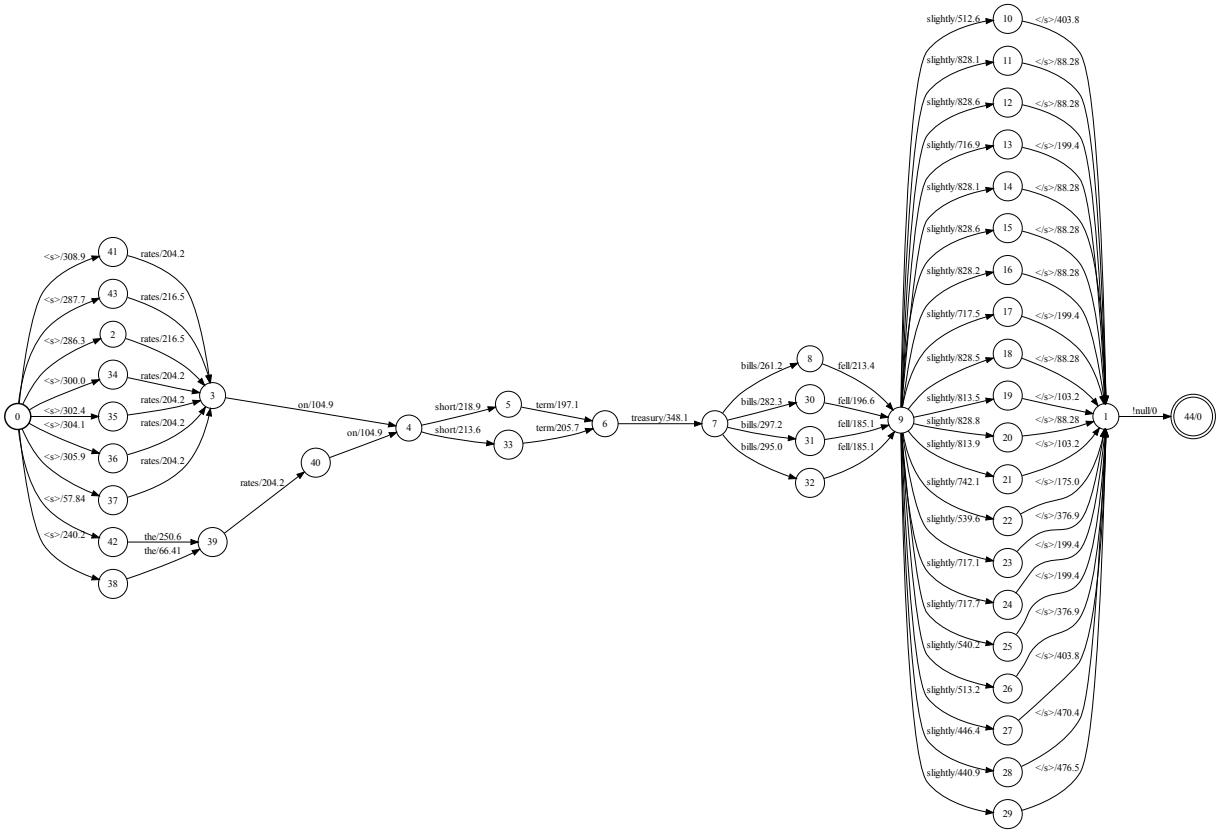

Figure 4.4: The arc-labeled FSM $\mathcal{F}_2$.

4.2.2 FSM Determinization

By looking closer into $\mathcal{F}_2$, one can discover that the FSM contains a high level of redundancy. Many arcs correspond to the same word with a different score. $\mathcal{F}_2$ is *nondeterministic* because, at a given state, there are different alternative

arcs with the same word label. Before parsing the converted FSM, it is essential to find an equivalent finite automata that is *deterministic* and that has as few nodes as possible. This way, the work necessary during parsing is reduced and efficient processing is ensured.

Any standard FSM can be *determinized* by the classical *determinization* algorithm. However, this principle does not apply to weighted FSMs. In general, a weighted finite automata that is not acyclic can be *determinized* (Mohri, 1997). Figure 4.5 depicts the equivalent FSM $\mathcal{F}_3$ obtained after applying the *determinization* algorithm given in Mohri and Riley (1997). $\mathcal{F}_3$ does not contain any redundancy and there is at most one arc labeled with a word at any node.

Figure 4.5: Equivalent FSM $\mathcal{F}_3$ obtained after *determinization*.

4.2.3 FSM Minimization

Like the weighted *determinization* algorithm, the classical finite state machine *minimization* algorithm is generalized to deal properly with the probabilities of alternative hypotheses (Mohri & Riley, 1997). Similar to standard FSMs, any *deterministic* and weighted FSM can be *minimized* (Mohri, 1997). $\mathcal{F}_4$ is

equivalent to $\mathcal{F}_2$ and $\mathcal{F}_3$ and it has the smallest number of nodes and the smallest number of arcs among all.

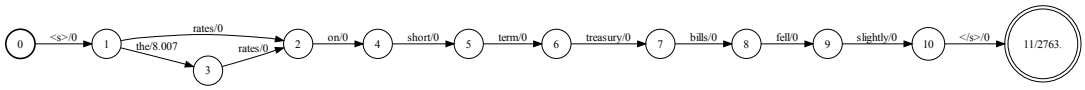

Figure 4.6: Equivalent FSM $\mathcal{F}_4$ obtained after *minimization*.

The weighted *minimization* algorithm works in two steps. In the first step, the weights are *pushed* toward the initial state as much as possible. The aim in this step is to obtain arcs with zero weights. In the second step, the arc label and the weight are combined as a new label in order to make use of the classical *minimization* algorithm.

4.2.4 N-best List Selection

The *minimization* process serves to shrink down the FSM to an equivalent automata with a suitable size for parsing. However, it is usually the case that the size is not small enough to meet the time and memory limitations in parsing. N-best list selection can be regarded as the last step in constricting the size. A subset of possible hypotheses is selected among many that are contained in the *minimized* FSM. The selection mechanism favors only the best hypotheses according to the scores present in the FSM arcs.

In Chow and Schwartz (1989), a Viterbi-style beam search algorithm that finds the most likely N pattern alternatives is introduced. The algorithm guarantees that the selected hypotheses are, in fact, the most likely sentence hypotheses. The exhaustive search algorithm presented in Tran et al. (1996) also makes use of Viterbi decoding. Likewise, the N-best path selection method implemented in Mohri et al. (1998) utilizes also Viterbi decoding. Figure 4.7 depicts the first-best hypothesis $\mathcal{F}_5$ that is extracted from $\mathcal{F}_4$. The path with the minimum weight is the first-best path.

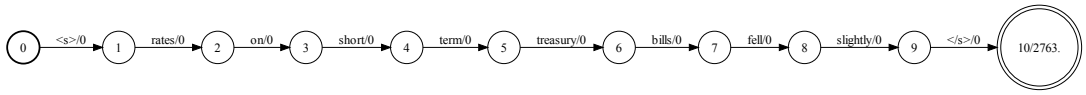

Figure 4.7: $\mathcal{F}_5$, first-best hypothesis selected from $\mathcal{F}_4$.

4.2.5 A^* Decoding

An alternative pruning method to the FSM approach explained in the previous sections is A^* decoding (Chelba & Jelinek, 1999). This method is based on the A^* search algorithm (Nilsson, 1971). In contrast to the FSM approach, the essence of the A^* decoder is defined as a structural language model (SLM). A broad introduction of the SLM is presented in Chelba and Jelinek (1998). The model assigns a probability to every hypothesis and to every possible binary parse tree.

Next, the A^* search algorithm is utilized to find the maximum scoring hypothesis by dynamic programming techniques. The whole approach is designed as a two-pass mode where, in the first pass, the hypotheses containing acoustic and language model scores are produced. In the second pass, the lattice accommodating the hypotheses is re-scored with A^* algorithm to find the best hypothesis. The structured language model is trained on a treebank corpus before the decoding process.

4.3 Word Graph Parsing

The essential idea behind the system proposed in this study is to initialize the chart of the MT parser using the simplified word graph. This way, all selected sentence hypotheses are processed simultaneously. The initialized chart is parsed until the first sentence hypothesis is selected. In its basic form, the chart models a confusion network which might lead to spurious parse trees. We extend the original chart representation and its processing in order to avoid spurious parses. The advantage of the approach lies essentially in its ability to rule out non-syntactic hypotheses in a parallel fashion.

The steps in word graph parsing is summarized below:

1. Initialize chart with the word graph
   (a) Calculate for each node the *distance* to the starting node
   (b) Calculate for each arc the *length*

(c) Create an edge for each arc

i. Starting position is equal to *distance*

ii. Span is equal to *length*

iii. Contains *start* and *end* node labels

iv. Contains *score*

2. Process the chart until no new edges can be created

3. At the end of parsing

(a) If parsing succeeds, return the edge with highest score

(b) If parsing fails, recover from the failure.

4.3.1 Chart Initialization

The chart initialization procedure CHART-INIT, listed in Algorithm 4.1, creates from an input FSM a valid chart that can be parsed in an active chart parser. The initialization starts with filling in the *distance* value for each node through the FILL-DISTANCE procedure given in Algorithm 4.2. The *distance* of a node in the FSM is defined as the number of arcs on the longest path from the start state to the current state. The *length* of an arc is defined as the difference of the *distance* values of the starting and ending nodes of the arc. After the *distance* and *length* values are set for all nodes and arcs in the FSM, an *edge* is created for each arc. The **while** loop in the CHART-INIT procedure passes over the arcs and copies the appropriate information onto the newly built edge. The *edge* structure also contains the *start* and *end* values in addition to the *weight* and *label* data fields. These

position values represent the edge location relative to the beginning of the chart. The starting and ending node information for the arc is also copied to the edge. This node information is later utilized in chart parsing to eliminate spurious parses. The newly created edge is inserted into the chart at each iteration of the **while** loop. The number of edges in the chart is equal to the number of edges in the input FSM.

Algorithm 4.1: The chart initialization procedure.

```
input: fsm
output: chart

procedure CHART–INIT(fsm)
    FILL–DISTANCE(fsm.start, 0)
    temp = fsm.arclist.first
    while temp is not null
        edge.label = temp.label
        edge.weight = temp.weight
        edge.start = temp.start.distance
        edge.finish = temp.finish.distance
        edge.arcstart = temp.start.id
        edge.arcfinish = temp.finish.id
        chart.edgelist.push(edge)
        temp = temp.next
    end while
end procedure
```

Consider the simple FSM $\mathcal{F}_6$ depicted in Figure 4.8, the corresponding two-dimensional chart and the related hypotheses. Using CHART-INIT algorithm, the chart is populated with the converted word graph before parsing begins. Words in the same column can

Algorithm 4.2: Procedure to find *distance* for each node in an FSM.

```
input: node, distance
output: updated node

procedure FILL-DISTANCE(node, distance)
    if node.distance < distance
        node.distance = distance
    end if
    temp = node.outarcs.first
    while temp is not null
        next = temp.next
        if next.distance < distance + 1
            next.distance = distance + 1
        end if
        next.visitor++
        if next.visitor == next.inarcs.size
            FILL-DISTANCE(next, node.distance + 1)
        end if
        temp = temp.next
    end while
end procedure
```

be regarded as a single lexical entry with different senses (e.g., '*boy*' and '*boycott*' in column 2). Words spanning more than one column can be regarded as idiomatic entries (e.g. '*escalated*' from column 3 to 5). Merged cells in the chart (e.g., '*the*' and '*yesterday*' at columns 1 and 6, respectively) are shared in both sentence hypotheses.

$\mathcal{F}_6$:

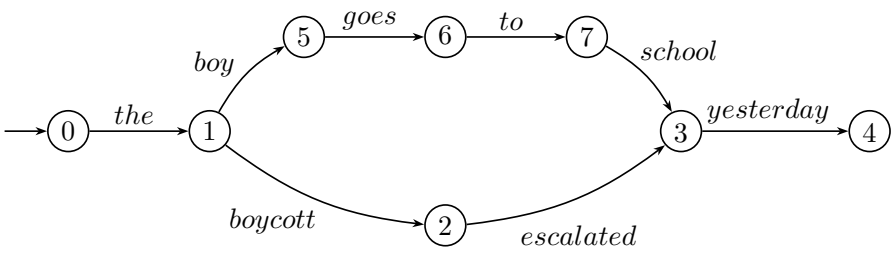

Chart:

0	1	2	3	4	5	6
$_0$ *the* $_1$	$_1$ *boy* $_5$	$_5$ *goes* $_6$	$_6$ *to* $_7$	$_7$ *school* $_3$	$_3$ *yesterday* $_4$	
	$_1$ *boycott* $_2$	$_2$ *escalated* $_3$				

Hypotheses:

- *The boy goes to school yesterday*
- *The boycott escalated yesterday*
- * *The boy escalated yesterday*

Figure 4.8: Sample FSM $\mathcal{F}_6$, the corresponding chart and the hypotheses.

4.3.2 Extended Chart Parsing

In a standard active chart parser, the chart depicted in Figure 4.8 could produce some spurious parses. For example, both of the complete edges in the initial chart at location [1-2] (i.e. '*boy*' and '*boycott*) can be combined with the word '*goes*', although '*boycott goes*' is not allowed in the original word graph. We have eliminated these kinds of spurious parses by making use of the *arcstart* and *arcfinish* values. These labels indicate

the starting and ending node identifiers of the path spanned by the edge in subject. The application of this idea is illustrated in Algorithm 4.3. Different from the previous implementation of BD-FUNDAMENTAL-RULE, the procedure has the additional parameters to define starting and ending node identifiers. Before creating a new incomplete edge, it is checked whether the node identifiers match or not.

When we consider the chart given in Figure 4.8, '$_1$ *boycott* $_2$' and '$_5$ *goes* $_6$' cannot be combined according to the new *fundamental rule* in a parse tree because the ending node id, i.e. 2, of the former does not match the starting node id, i.e. 5, of the latter. In another example, '$_0$ *the* $_1$' can be combined with both '$_1$ *boy* $_5$' and '$_1$ *boycott* $_2$' because their respective node identifiers match. After the two edges, '*boycott*' and '*escalated*', are combined and a new edge is generated, the starting node identifiers for the entire edge will be as in '$_1$ *boycott escalated* $_3$'. A small fragment of the chart at the end of parsing is depicted in Figure 4.9.

The utilization of the node identifiers enables the two-dimensional modeling of a word graph in a chart. This extension to chart parsing makes the current approach word-graph based rather than confusion-network based. Parse trees that conflict with the input word graph are blocked and all the processing resources are dedicated to proper edges.

Another important extension as presented in Algorithm 4.3 is the utilization of the ASR scores. Each edge has a *weight* that comes from the initial lattice. Whenever a new edge is created during parsing, its *weight* score is assigned the sum of the compounding edge scores. The scores in the edges are used at the end of parsing to select the most feasible edge. For example, if there is more than one edge with the final symbol category, than the one having the highest weight is selected as the succeeding edge. The time

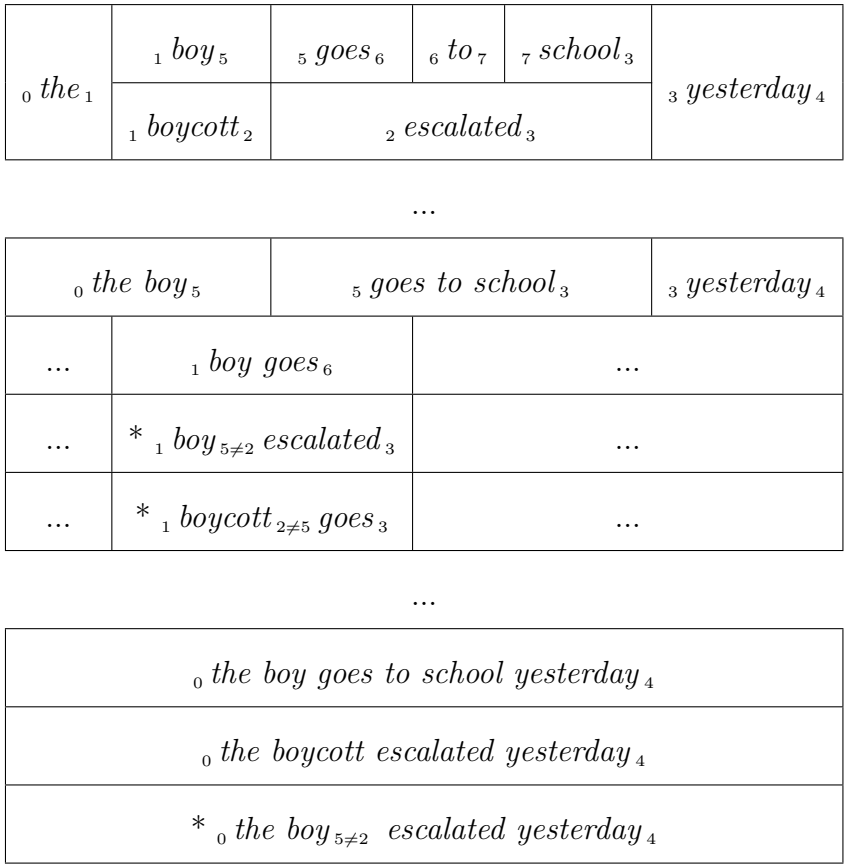

Figure 4.9: A sample fragment of the chart at the end of parsing. Edges marked with a star are not built at all.

complexity of the algorithm is $O(n^3)$, which is similar to the one of the regular chart parsing algorithm.

4.3.3 Parse recovery

As described in Section 3.3.4, parse recovery process takes control if no successful parse tree can be constructed. Trying to parse a word graph instead of a stream of words already introduced some complexities into the chart parsing process. Similarly, word graphs bring up new difficulties into the parse recovery process. The original parse

Algorithm 4.3: Updated fundamental rule to parse a word graph.

```
procedure BD–Fundamental–Rule (A → B • α • C, [j,k], [n_s,n_e], w_e)
    if B = βD          // edge is incomplete
        for each (D → •δ•, [i,j], [n_r,n_s], w_c) in chart
            Push (agenda, (A → β • DαC, [i,k], [n_r,n_e], w_e + w_c))
        end for
    end if
    if C = Dγ          // edge is incomplete
        for each (D → •δ•, [k,l], [n_e,n_f], w_c) in chart
            Push (agenda, (A → B • αDγ, [j,l], [n_s,n_f], w_e + w_c))
        end for
    end if
    if B is null and C is null // edge is complete
        for each (D → βA • γ • δ, [k,l], [n_e,n_f], w_i) in chart
            Push (agenda, (D → β • Aγ • δ, [j,l], [n_s,n_f], w_e + w_i))
        end for
        for each (D → β • γ • Aδ, [i,j], [n_r,n_s], w_i) in chart
            Push (agenda, (D → β • γA • δ, [i,k], [n_r,n_e], w_e + w_i))
        end for
    end if
end procedure
```

recovery process can produce a sequence of words that does not belong to the set of hypotheses that are represented by the initial word graph. As in the parsing stage, this obstacle originates from the two dimensional structure of the word graph.

Regarding the word graph depicted in Figure 4.8, '*the boycott goes to the school yesterday*' is not part of the input hypotheses. However, this sentence can be generated at the end of the recovery process if the final chart contains the edges '*the boycott*' and '*goes to the school yesterday*'. To avoid these kind of spurious parses, the idea

of start and end node identifiers will be used as in parsing. The node identifiers in '$_0$*the boycott*$_2$' and '$_5$*goes to the school yesterday*$_4$', i.e. 2 and 5, do not match and the spurious hypothesis is not generated.

Weights associated to edges are also used in the parse recovery process. Whenever there is more than one possible edge to include in the parse tree than the one that has the highest weight is included in the final parse tree.

4.4 N-best word graph vs. N-best list

There are two main differences between parsing a word graph and parsing a list of sentences. The first difference arises from the existence of scores in arcs of the word graph. Exploiting the scores moves the entire parsing process to a hybrid level. This benefit is discussed in previous sections in detail. The second deviation originates from the fact that multiple sentences are parsed at the same time by using the word graph. Unlike parsing a single sentence at a time as in N-best lists, the whole word graph is parsed in parallel. In this part, we comment on the parallel parsing mechanism of the N-best word graph approach.

The most important benefit of the parallel parsing approach is that it processes shared edges one time only. This fact introduces an important benefit in the number of created edges during parsing. In the N-best list approach, every word in every hypothesis is inserted into the chart. If a word occurs in all the hypothesis then it is inserted to the chart N times as a leaf node. Consequently, the edges that are derived from the leaf node are reproduced N times. This reproduction causes an excess in the number of edges. Identical edges are created at each instance of list parsing in contrast

with the word graph parsing. Thus, we can visualize the N-best word graph approach as an optimization to the N-best list approach. All the edges that are placed into the chart are unique edges in lattice parsing. No other edge with the same features are ever created.

CHAPTER 5

ENHANCEMENT OF THE WORD GRAPH

In standard and traditional ASR systems, acoustic model (AM) and language model (LM) alone appoint the system architecture. The output word graph produced by these standard ASR systems contain scores associated with AM and LM only. Other applications that make use of this word graph utilize these two scores. In this chapter, we elaborate on how the word graph can be enhanced further to include different model scores.

5.1 Prosody in Speech Processing

ASR systems that are composed of AM and LM alone ignore high level prosodic information that is present in the utterance. Prosody is defined as the rhythmic and intonational aspect of an utterance. Although it is one of the most well-studied features in ASR, it is not captured in most large vocabulary continuous speech recognition (LVCSR) systems (Shriberg & Stolcke, 2002). However, the interest to use prosody in speech tasks is increasing (Ostendorf, Shafran, & Bates, 2003). Humans use prosody ex-

tremely in everyday communication because it provides valuable information to disambiguate meaning. Speech that is cleared from natural prosody requires higher cognitive load (Chen & Hasegawa-Johnson, 2003).

Low level features in speech, e.g. duration, pitch accents and boundary tones, F0, voicing, energy and spectral tilt form the prosody and it is closely related with the syntax and semantics of the speech. In general, prosodic features are extracted directly from the speech signal and from the output of an automatic speech recognizer.

There are different challenges in the integration of prosody into ASR systems. First, processing of prosodic features, e.g. extracting and normalizing, should be done automatically. Next, the proposed model should be able to tolerate errors. Finally, it must be feasible to use in different speech applications.

In Shriberg and Stolcke (2004), an approach is described to use prosody in various speech related tasks: Structural tagging (e.g. finding sentence boundaries and disfluencies), pragmatic and paralinguistic tagging (e.g. classifying dialog acts, emotion etc.), speaker recognition and word recognition. Kim and Woodland (2001) incorporate prosodic information with acoustic and language model information to create a combined system for punctuation generation and speech recognition.

Chen and Hasegawa-Johnson (2003) present a novel approach that improves robustness by leveraging the dependence between prosody and syntax. The presented model describes the joint probability distribution of concurrent word and prosody sequences. In Szaszak and Vicsi (2007), the impact of using prosodic features in the recognition of agglutinating and fixed stress languages is investigated. Ananthakrishnan and Narayanan (2007) introduce a system which includes prosody-enriched word graphs. Syllable level lattices are generated by a standard ASR and later enriched with

prosodic information. 2% of relative improvement is claimed to be obtained in syllable error rate. Syllable and related sub-word unit recognition is used in tasks such as name recognition. Speech recognizers that produce word hypotheses perform poorly on tasks which have to account for out-of-vocabulary words.

Not only in speech recognition is intonation information important, but it also plays significant role in the speech synthesis task. In Prevost and Steedman (1994), a model for generating prosodically appropriate speech synthesis is presented. The authors demonstrate the ability of the proposed model to generate a variety of intonational possibilities depending on the discourse context. Other studies presenting the utilization of prosody in speech synthesis task is widely available in literature.

5.2 The Prosody Model

Prosodic model has to be combined with other knowledge sources in order to use it in speech recognition tasks. A common approach is to integrate it with lexical information (Shriberg & Stolcke, 2004). Prosodic modeling is tied to LM to improve disambiguation efficiency.

In Chen and Hasegawa-Johnson (2003), the task of speech recognition is reduced to find $W = (w_1, \ldots, w_M)$, the sequence of word labels, that maximize the recognition probability given in Equation 5.1. The basic idea in the equation is to condition the language model on prosody.

$$[\tilde{W}] = argmax\ p(O|W,P)\ p(W,P) \tag{5.1}$$

$$= argmax\ p(O|Q,H)\ p(Q,H|W,P)\ p(W,P) \tag{5.2}$$

In Equation 5.1 and Equation 5.2, $P = (p_1, \ldots, p_M)$ is the sequence of prosody labels, $O = (o_1, \ldots, o_T)$ is the sequence of observed acoustic feature vectors, $Q = (q_1, \ldots, q_L)$ is the sequence of sub-word units, $H = (h_1, \ldots, h_L)$ is the sequence of discrete "hidden mode" vectors. In this probabilistic model, w_m and p_m together denote the prosody dependent word label and q_l and h_l denote the allophone label. In Equation 5.2, $p(O|Q, H)$ component represents the acoustic model, $p(Q, H|W, P)$ represents the pronunciation model and $p(W, P)$ represents the language model.

Prosody dependent N-gram language modeling requires a large amount of prosodically transcribed data. In Chen and Hasegawa-Johnson (2003), prosody-syntax dependence is utilized to diminish the data sparseness.

5.3 Prosody and Word graph

In various systems that make use of prosodic information, the word graph is enriched after a baseline recognizer produces the lattice including only AM and LM scores. For example, the syllable-level time-state alignments are used in Ananthakrishnan and Narayanan (2007) to extract acoustic-prosodic features which function as a binary classifier, i.e. presence vs. absence of pitch accent. Each arc in the final lattice contains the binary prosody information as depicted in Figure 5.1.

Once the word graph is enriched with prosodic information, processing in following stages should make use of this information. In the case of syntactic parsing, a prosody-aware grammar is required. That is, the parsing grammar should be able to make use of the prosodic information. Otherwise the enhancement of the word graph does not have any impact on the results.

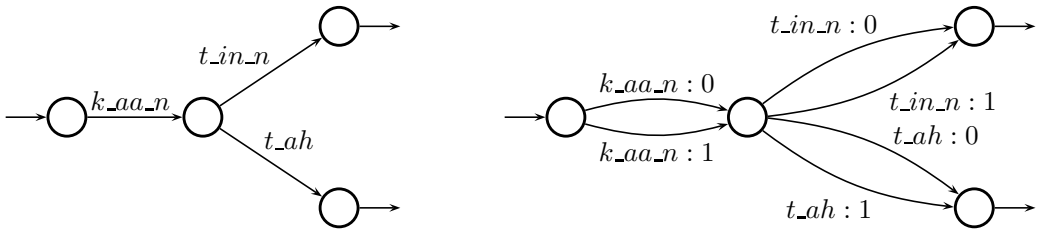

Figure 5.1: Baseline word graph and enriched word graph (Ananthakrishnan & Narayanan, 2007).

5.4 Prosody and Syntax

The suprasegmental nature of prosodic features is a problem in developing a combined model to process prosody and syntax together. Prosodical units and syntactic units do not share always the same boundaries. However, there has been attempts to integrate phonological structures into different syntactic frameworks as in Butt and King (1998) and Steedman (2000). Steedman develops a new semantics for intonation structure that is fully integrated into Combinatory Categorial Grammar (CCG). Butt and King suggest a new phonological component called *p-structure* in order to integrate phonological representations into Lexical Functional Grammar (LFG). This new addition is based on LFG's projection architecture. In brief, the p-structure is constructed by domains corresponding to the prosodic hierarchy. A sample p-structure representation is shown in Figure 5.2.

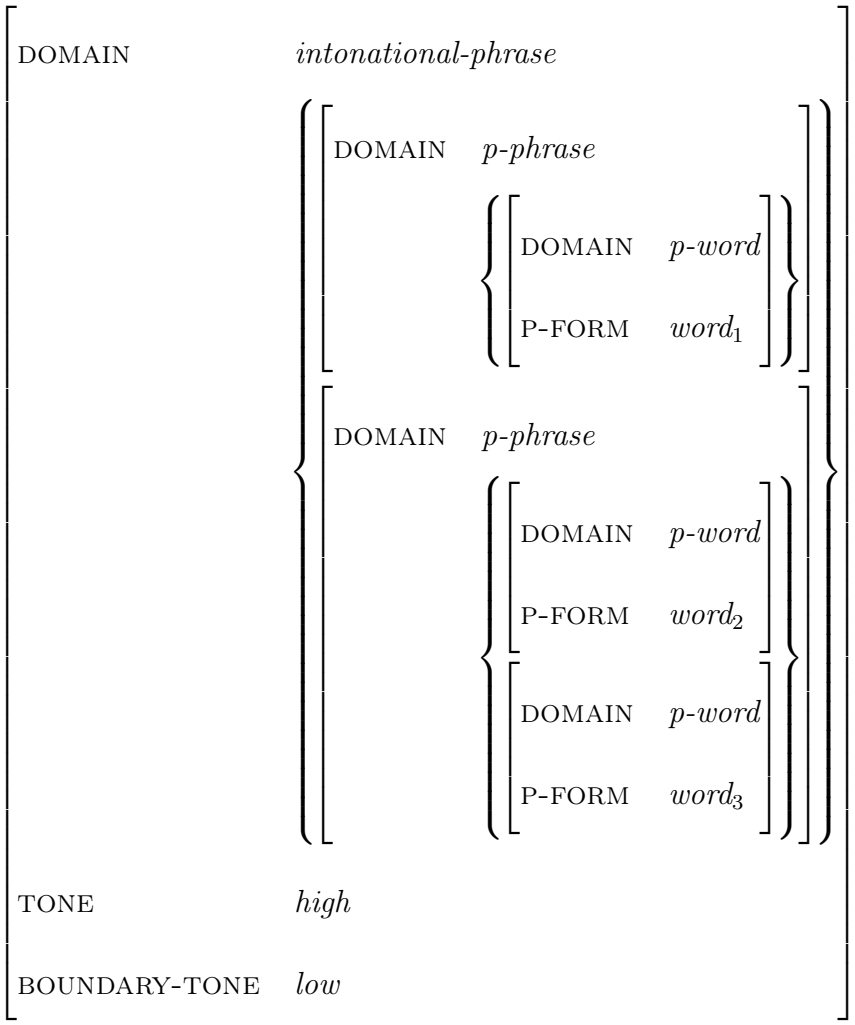

Figure 5.2: A sample representation of the p-structure taken from Butt and King (1998) for a three word intonational phrase.

CHAPTER 6

EXPERIMENTS AND EVALUATION

In this chapter, we present the experiments from different perspectives in order to evaluate the proposed system. The aspects of any evaluation should be closely related to the task that the system is working on. In translation, accuracy and fluency are important aspects that provide information about the conformity of the result to the correct solution. In addition, the method by which the inspection is accomplished, e.g. whether manually or automatically, is also important.

Another aspect in evaluation is the decision of whether to assess the system as a whole or each component separately. Evaluating the performance of the total application from beginning to end is called *extrinsic* evaluation. The opposite is the component-based evaluation, which is defined as *intrinsic*. Extrinsic evaluation is more expensive compared to intrinsic evaluation since the former requires the involvement of all ingredients in a system. However, it is important to verify that an intrinsic improvement yields an extrinsic improvement. In this section, we conduct intrinsic and extrinsic experiments to put forth the gains of our approach based on standard evaluation metrics.

6.1 Metrics

The metrics established for NLP tasks generally try to estimate the closeness of the results to the solution. Different NLP tasks use different metrics to evaluate the system performance. Recall and precision values are used in evaluating information extraction systems. Speech recognition systems are mainly evaluated using the word error rate (WER) metric, which reports how much the proposed string differs from a reference transcription. WER can be calculated as given in Equation 6.1, where $\mathcal{S}$ is the number of substitutions, $\mathcal{D}$ is the number of the deletions, $\mathcal{I}$ is the number of the insertions and $\mathcal{N}$ is the number of words in the reference text.

$$WER = \frac{\mathcal{S} + \mathcal{D} + \mathcal{I}}{\mathcal{N}} \tag{6.1}$$

Metrics for the translation task give details about the closeness of the output translation to one or more available reference translations. One of the most popular methods in the automatic evaluation of MT is BLEU (Papineni et al., 2002). As mentioned before, it requires at least one human translation for comparison. The idea is to use the weighted average of variable length phrase matches in the reference translation. The more common patterns that exist between two translations, the higher the BLEU score. This fact makes the statistical MT systems more advantageous than rule-based MT systems, because statistical systems are trained on aligned corpora with a pattern-based approach. Similar to BLEU, the NIST metric (Doddington, 2002) also uses an N-gram co-occurrence-scoring approach. There are some other methods proposed to overcome the proved weakness of the BLEU and NIST methods. The METEOR evaluation method (Banerjee & Lavie, 2005; Lavie & Agarwal, 2007) makes two main additions to

the BLEU method. It utilizes a stemmer and a dictionary of synonyms to get rid of the strict matching approach. WordNet (Miller & Fellbaum, 2008) is used in METEOR to query the stem and synonyms. In the output translation, if a phrase contains the same stem with a different surface form, or if it contains a synonym, the translation still gets a score. In BLEU and NIST, stems and synonyms are not taken into consideration. Currently, METEOR is available for English, French, German, Spanish and Czech only.

Another automatic evaluation method, different from the previous ones because it does not rely directly on co-occurrences, is the TER method (Snover et al., 2006). TER, which is inspired by the WER metric, is calculated with the minimum number of edits required to modify the output translation so that it exactly matches one of the reference translations.

It is important to mention that all the discussed automatic evaluation methods show a strong correlation between the obtained scores and human assessments of translation quality. In our experiments, we present WER and BLEU scores to evaluate the system. We do not present METEOR scores as it is not available for the target language (i.e. Arabic) we have chosen for the experiments.

6.2 Experimental Setup

The experiments carried out in this thesis are run on word graphs based on 1993 benchmark tests for the ARPA spoken language program (Pallett et al., 1994). In the large-vocabulary continuous speech recognition (CSR) tests reported by Pallett et al. (1994), Wall Street Journal-based CSR corpus material was made use of. The tests intended to measure basic speaker-independent performance on a 64K-word read-speech

test set which consists of 213 utterances. Each of the 10 different speakers provided from 20 to 23 utterances. An acoustic model and a trigram language model is trained using Wall Street Journal data by Chelba (2000) who also generated the 213 word graphs used in the current experiments. The word graphs, referred as HUB-1 data set, contain both the acoustic scores and the trigram language model scores. Previously, the same data set was used in other studies (Chelba, 2000; Roark, 2001; Hall, 2005) for language modeling task in ASR.

6.2.1 N-best list pruning

The 213 word graphs in the HUB-1 data set are pruned as described in Section 4.2 in order to prepare them for chart parsing. AT&T toolkit (Mohri et al., 1998) is used for determinization and minimization of the word graphs and for n-best path extraction. Prior to feeding in the word graphs to the FSM tools, the acoustic model and the trigram language model in the original lattices are combined into a single score using Equation 6.2, where $\mathcal{S}$ represents the combined score of an arc, $\mathcal{A}$ is the acoustic model (AM) score, $\mathcal{L}$ is the language model (LM) score, α is the AM scale factor and β is the LM scale factor.

$$\mathcal{S} = \alpha\,\mathcal{A} + \beta\,\mathcal{L} \tag{6.2}$$

Figure 6.1 depicts the word error rates for the first-best hypotheses obtained by using $\alpha = 1$ and β values from 1 to 25. The lowest WER (13.32) is achieved when α is set to 1 and β to 15. This result is close with the findings from Hall (2005) who reported to use 16 as the LM scale factor for the same data set. WER score for LM-only was 26.8 where in comparison the AM-only score was 29.64. The results imply that

the language model has more predicting power over the acoustic model in the HUB-1 lattices. For the rest of the experiments, we used 1 and 15 as the acoustic model and language model scale factors, respectively.

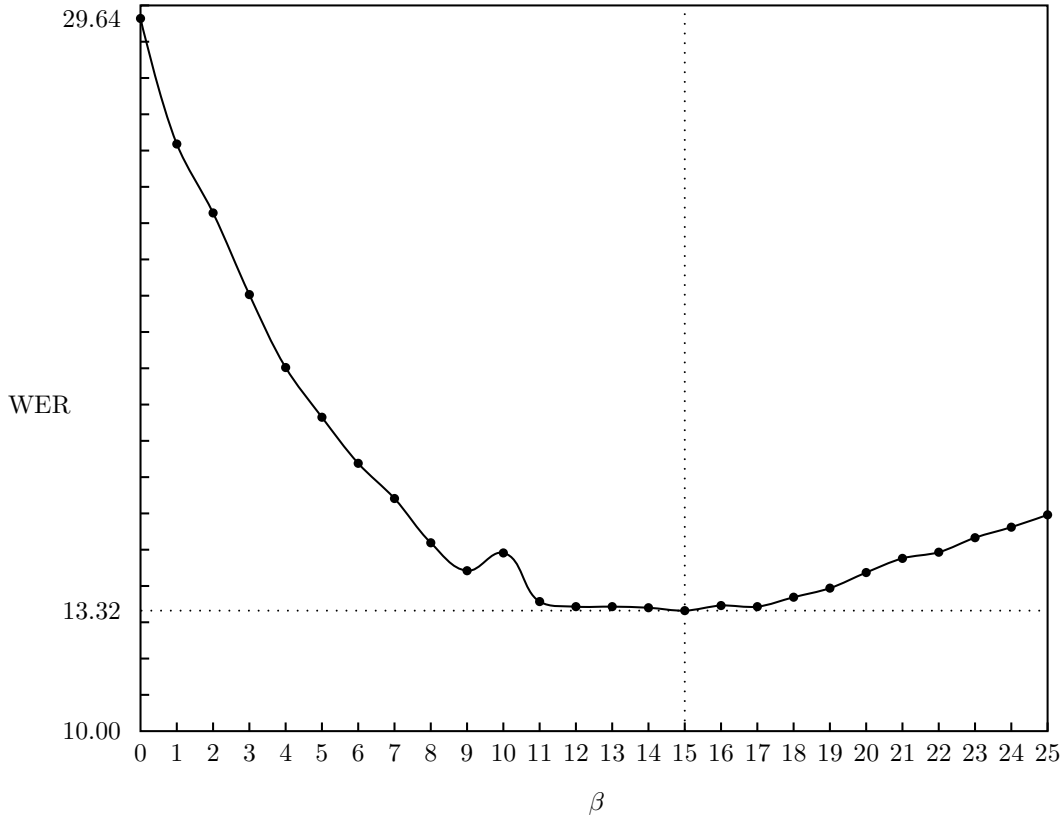

Figure 6.1: WER for HUB-1 first-best hypotheses obtained using different language-model scaling factors and $\alpha = 1$.

6.2.2 Word graph accuracy

Using the scale factors found in the previous section we built N-best word graphs for different N values. In order to measure the word graph accuracy we constructed the FSM for reference hypotheses, $\mathcal{F}_{Ref}$, and we took the intersection of all the word graphs with the reference FSM. Table 6.1 lists the word graph accuracy rate for different N

Table 6.1: Word graph accuracy for different N values in the data set with 213 word graphs.

N	Accuracy
1	30.98
10	51.17
20	56.34
30	58.22
40	59.15
50	59.15
60	59.15
70	59.15
80	59.15
90	60.10
100	60.10
full	66.67

values. For example, an accuracy rate of 30.98 denotes that 66 word graphs out of 213 contain the correct sentences. The accuracy rate for the original word graphs in the data set is 66.67 which indicates that only 142 out of 213 contain the reference sentence. In 71 of the instances, the reference sentence is not included in the untouched word graph. The accurate rates express the maximum sentence error rate (SER) that can be achieved for the data set.

6.2.3 Linguistic Resources

The English grammar used in the chart parser contained 20 morphology analysis rules and 225 syntax analysis rules. All the rules and the unification constraints were implemented in LFG formalism as part of an MT project. The number of rules to model the language grammar is quite few compared to probabilistic CFGs which contain more than 10 000 rules. The monolingual analysis lexicon consists of 40 000 lexical entries. The English to Arabic bilingual transfer lexicon also contains 40 000 entries.

6.3 Chart parsing experiments

We conducted experiments to compare the performance of N-best list parsing and N-best word graph parsing. Compared to the N-best list approach, in N-best word graph parsing approach, the shared edges are processed only once for all hypotheses. This saves a lot on the number of complete and incomplete edges generated during parsing. Hence, the overall processing time required to analyze the hypotheses are reduced. In an N-best list approach, where each hypothesis is processed separately in the analyzer, there are different charts and different parsing instances for each sentence hypothesis. Shared words in different sentences are parsed repeatedly and same edges will be created at each instance.

Table 6.2 represents the number of complete and incomplete edges generated for the NIST HUB-1 data set. For each hypothesis, 164 complete edges and 2490 incomplete edges are generated in the N-best list approach. In the N-best word graph approach, the average number of complete edges and incomplete edges reduced to 31 and 341, respectively. The decrease is 81.1% in complete edges and 86.3% in incomplete edges

Table 6.2: Number of complete and incomplete edges generated for the NIST HUB-1 data set for N-best list and N-best word graph approaches.

Approach	Hypotheses	Complete edges	Incomplete edges
N-best list	4869	798 K	12.125 M
	1	164	2490
N-best	4869	150.8 K	1.662 M
word graph	1	31	341

Table 6.3: Number of complete edges generated for the NIST HUB-1 data set using different approaches.

Approach	Edges
N-best word graph	150.8 K
PCFG parser	880.9 K
Charniak Parser	2,950.7 K

for the NIST HUB-1 data set. The profit introduced in the number of edges by using the N-best word graph approach is immense.

The comparison of the number of complete edges produced for the same data set in the PCFG parser (Hall & Johnson, 2003), Charniak parser (Charniak, 2001) and our approach is given in Table 6.3. As expected, our unification-based active chart parser produces the least number of edges because of its unification and functional expression mechanism.

6.4 Language modeling experiments

In this part of the experiments we tested the proposed approach from a language modeling perspective. The aim of this experiment is to test how the N-best word graph approach can be used as the LM component in an ASR system. Table 6.4 reports the WER results of different language models. The other scores presented in Table 6.4 are taken from Hall (2005). NIST HUB-1 data set is used in all of the results.

The Charniak parser (Charniak, 2001) scores the best result (11.8) in the language modeling tests. Multi stage PCFG parsing complemented with *attention shifting* (Hall & Johnson, 2004) scores the second best result. Our approach, in which the ASR lattice is preprocessed and parsed with manually created rules scores 12.2. Five other models score below our result. The quantitative result obtained in this experiment shows that parsing the N-best word graph using a unification-based active chart parser does a decent job. This proves that we can support the LM component in an ASR system with the parsing engine of a rule-based MT system.

6.5 Machine Translation experiments

The aim of this experiment was to evaluate the proposed approach from the MT perspective. The NIST HUB-1 data set is used as in the previous experiments. The 213 hypothesis that are obtained in the first-best approach and the N-best word graph approach are utilized as the source text in the MT experiment. We used a freely and publicly available MT system for our purpose. Our objective in this MT experiment is not to maximize the scores, instead, we want to figure out the impact of the WER on the MT results. Therefore, we did not apply any normalization to the input text (e.g.

Table 6.4: WER for different language models on the NIST HUB-1 data set including N-best word graph parsing approach that is presented in this study.

Model	WER
Charniak Parser (Charniak, 2001)	11.8
Attention Shifting (Hall & Johnson, 2004)	11.9
PCFG (Hall, 2005)	12.0
N-best word graph *(this study)*	**12.2**
A* decoding (Xu, Chelba, & Jelinek, 2002)	12.3
PCFG (Roark, 2001)	12.7
PCFG (Hall & Johnson, 2004)	13.0
40m-word trigram (Hall & Johnson, 2003)	13.7
PCFG (Hall & Johnson, 2003)	15.5

Table 6.5: BLEU and NIST scores for first-best and N-best word graph approaches.

Model	BLEU	NIST
First-best	0.0285	2.2888
N-best word graph	0.0289	2.2969

converting *"does n't"* to *"doesn't"*).

The BLEU and NIST scores for the two different approaches are reported in Table 6.5. The slight improvement in WER from 13.3 to 12.2 as reported in the language modeling experiments are also reflected in the scores. Between the first-best approach and the N-best word graph approach, there is a 1.4% improvement in terms of the BLEU scores and 0.4% improvement in terms of the NIST scores. We can conclude that there is a linear relation between WER in language modeling and BLEU/NIST score in MT.

6.6 Evaluation

In this section we compare the different approaches from a qualitative perspective. Table 6.6 lists the different approaches and their properties. All the approaches listed in Table 6.6 have statistical components. Except the trigram approach, all others also utilize structural techniques. Our approach and Hall's approaches (Hall & Johnson, 2004; Hall, 2005) include some preprocessing steps in order to prune the ASR lattice before parsing.

Different from all other systems, our approach utilizes a unification-based active chart parser and manually created grammar rules. This feature enables the approach to be applied as a perfect integrator between the ASR and a rule-based MT system. Other approaches are designed to be used only as a language model. The N-best word graph approach, however, is used at the same time as a language model and as an analysis component in a rule-based MT system.

Table 6.6: Evaluation of the different language modeling approaches for speech translation.

Model	Coverage	Pruning	Parser	Unification-based
Charniak Parser (Charniak, 2001)	ASR		Charniak parser	
Attention Shifting (Hall & Johnson, 2004)	ASR	√	Chart parser	
PCFG (Hall, 2005)	ASR	√	Chart parser	
N-best word graph (this study)	ASR, MT	√	Chart parser	√
A* decoding (Xu et al., 2002)	ASR		*NA*	
PCFG (Roark, 2001)	ASR		PCFG	
PCFG (Hall & Johnson, 2004)	ASR		Chart parser	
40m-word trigram (Hall & Johnson, 2003)	ASR		*NA*	
PCFG (Hall & Johnson, 2003)	ASR		Chart parser	

CHAPTER 7

CONCLUSIONS AND FUTURE DIRECTIONS

The primary aim of this research was to develop a new and efficient method for integrating an ASR system with a rule-based MT system. The innovative idea is to populate the chart parser inside the MT analyzer with the word graph that comes out of the ASR component. This way, structural language modeling task in ASR and the analysis task in MT are combined. This combination has enabled the usage of legacy rule-based MT systems in speech translation task to be as effective as statistical MT systems. Language modeling task is achieved because the parse tree that is generated at the end of lattice parsing belongs to the winning hypothesis. On the other hand, analysis task is achieved because the parse tree is also associated with an *attribute value matrix* to be used in later stages of the MT task. We also utilize the statistical information that comes from the ASR inside the chart parser. This utilization hybridizes the system and it supplements the process of resolving the ambiguity in syntactic analysis.

We present in this thesis an attempt to blend statistical ASR systems with rule-based MT systems. The objective of the tight assembly of these two components

was to obtain an enhanced ST system. This approach can be generalized to any MT system employing chart parsing in its analysis stage. In addition to utilizing rule-based MT in ST, this study also used word graphs and chart parsing with new extensions. Specifically, we expand the regular chart parsing algorithm to parse the speech lattice while eliminating spurious parses.

We call the newly developed coupling approach as *N-best word graph.* We have shown in our experiments that this approach performs better than the alternative *first-best* and *N-best list* approaches. Before our new approach, all rule-based MT systems were utilized inside an ST system either using the *first-best* method or the *N-best list* method.

The architectural design and details of the devised ST system are explored in Chapter 3. Elements of the MT system are provided including the morphological analysis, parsing, transfer and generation. In Chapter 4, we explored the particular stages of word graph processing in detail. The word graph is minimized and pruned before it is being parsed. Later in the same chapter, we provided the details of word graph parsing using a unification-based active chart parser. We also elaborated on further enhancements of the word graph in Chapter 5. An ordinary ASR system generates a lattice that contains only acoustic and language model scores. We have researched how other prosodic information can be integrated into the ASR lattice.

The experiments described in this thesis show that parsing the word graph at one instance improves the translation performance, compared to parsing all sentence hypotheses separately. In terms of the number of complete edges, N-best word graph approach produces 81.1% less edges during parsing if compared to the N-best list approach. We also conducted language modeling experiments and compared our scores

to other scores available in the literature based on NIST HUB-1 data set. The scores obtained in this study prove that structural approaches are as competitive as statistical approaches.

For further improvement of the ST system, our future studies include the following:

1. Preprocessing the pruned word graph before parsing to improve the language modeling capability of the system.

2. Extension of the optimization capabilities of the parser to further improve the efficiency of the chart parser.

3. Implementing the transfer and generation modules to achieve a complete MT system. Similar to the analysis module, transfer and generation modules should also utilize statistical tools in order to achieve a fully hybrid MT system.

4. Enhancing the ASR word graph with prosody information and using it in the disambiguation process inside the syntactic analyzer. Forwarding the prosody information from MT toward the TTS to utilize it in speech generation.

BIBLIOGRAPHY

Aho, A. V., Sethi, R., & Ullman, J. D. (1986). *Compilers: Principles, Techniques, and Tools*. Addison-Wesley.

Ananthakrishnan, S., & Narayanan, S. (2007). Prosody-enriched lattices for improved syllable recognition. In *Interspeech 2007*, pp. 1813–1816.

Arranz, V., Comelles, E., Farwell, D., Nadeu, C., Padrell, J., Febrer, A., Alexander, D., & Peterson, K. (2004). A speech-to-speech translation system for Catalan, Spanish, and English. In *AMTA*, pp. 7–16.

Banerjee, S., & Lavie, A. (2005). Meteor: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the 43th Annual Meeting of the ACL'05*. ACL.

Bertoldi, N., & Federico, M. (2005). A new decoder for spoken language translation based on confusion networks. In *Automatic Speech Recognition and Understanding Workshop (ASRU)*.

Blache, P., & Morin, J.-Y. (1990). Bottom-up filtering: a parsing strategy for GPSG. In *Proceedings of the 13th conference on Computational linguistics*, pp. 19–23 Morristown, NJ, USA. Association for Computational Linguistics.

Boitet, C., & Seligman, M. (1994). The "Whiteboard" architecture: A way to integrate heterogeneous components of NLP systems. In *COLING*, pp. 426–430.

Bresnan, J. (1982). Control and complementation. In Bresnan, J. (Ed.), *The Mental Representation of Grammatical Relations*, pp. 282–390. MIT Press, Cambridge, MA.

Brown, P. F., Cocke, J., Pietra, S. D., Pietra, V. J. D., Jelinek, F., Lafferty, J. D., Mercer, R. L., & Roossin, P. S. (1990). A statistical approach to machine translation. *Computational Linguistics*, *16*(2), 79–85.

Butt, M., & King, T. H. (1998). Interfacing phonology with lfg. In *Proceedings of the LFG98 Conference*.

Chappelier, J.-C., Rajman, M., Aragues, R., & Rozenknop, A. (1999). Lattice parsing for speech recognition. In *TALN'99*, pp. 95–104.

Charniak, E. (2001). Immediate-head parsing for language models. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics.

Chelba, C. (2000). *Exploiting Syntactic Structure for Natural Language Modeling*. Ph.D. thesis, Johns Hopkins University.

Chelba, C., & Jelinek, F. (1998). Exploiting syntactic structure for language modeling. In *Proceedings of the 36th annual meeting on Association for Computational Linguistics*, pp. 225–231 Morristown, NJ, USA. Association for Computational Linguistics.

Chelba, C., & Jelinek, F. (1999). Structured language modeling for speech recognition. In *Proceedings of NLDB99* Klagenfurt, Austria.

Chen, K., & Hasegawa-Johnson, M. (2003). Improving the robustness of prosody dependent language modeling based on prosody syntax dependence. In *Proc. of IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU '03*, pp. 435–440.

Chien, L.-F., Chen, K. J., & Lee, L.-S. (1990a). An augmented chart data structure with efficient word lattice parsing scheme in speech recognition

applications. In *Proceedings of the 13th conference on Computational linguistics*, pp. 60–65 Morristown, NJ, USA. Association for Computational Linguistics.

Chien, L.-F., Chen, K. J., & Lee, L.-S. (1990b). An augmented chart parsing algorithm integrating unification grammar and markov language model for continuous speech recognition. In *ICASSP 90: Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pp. 585–588. IEEE.

Chien, L.-F., Chen, K. J., & Lee, L.-S. (1993). A best-first language processing model integrating the unification grammar and markov language model for speech recognition applications. *IEEE Transactions on Speech and Audio Processing*, *1*(2), 221–240.

Chow, Y.-L., & Roukos, S. (1989). Speech understanding using a unification grammar. In *ICSLP'89: Proc. of the International Conference on Acoustics, Speech, and Signal Processing*, pp. 727–730. IEEE.

Chow, Y.-L., & Schwartz, R. (1989). The N-best algorithm: an efficient procedure for finding top n sentence hypotheses. In *HLT'89: Proceedings of the workshop on Speech and Natural Language*, pp. 199–202 Morristown, NJ, USA. Association for Computational Linguistics.

Clarkson, P., & Rosenfeld, R. (1997). Statistical language modeling using the CMU–cambridge toolkit. In *Proc. Eurospeech '97*, pp. 2707–2710 Rhodes, Greece.

Collins, C. (2004). Head-driven probabilistic parsing for word lattices. Master's thesis, University of Toronto.

Doddington, G. (2002). Automatic evaluation of machine translation quality using N-gram co-occurence statistics. In *Proc. International Conference on Human Language Technology Research*, pp. 138–145. Morgan Kaufmann Publishers Inc.

Dyer, C., Muresan, S., & Resnik, P. (2008). Generalizing word lattice translation. In *Proceedings of ACL-08: HLT*, pp. 1012–1020 Columbus, Ohio. Association for Computational Linguistics.

Earley, J. (1970). An efficient context-free parsing algorithm. *Communications of the ACM*, *13*(2), 94–102.

Good, I. J. (1953). The population frequencies of species and the estimation of population parameters. *Biometrika*, *40*(3/4), 237–264.

Hall, K. (2005). *Best-First Word Lattice Parsing: Techniques for Integrated Syntax Language Modeling*. Ph.D. thesis, Brown University.

Hall, K., & Johnson, M. (2003). Language modelling using efficient best-first bottom-up parsing. In *ASR'03: IEEE Workshop on Automatic Speech Recognition and Understanding*, pp. 507–512. IEEE.

Hall, K., & Johnson, M. (2004). Attention shifting for parsing speech. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, p. 40 Morristown, NJ, USA. Association for Computational Linguistics.

Harper, M., Jamieson, L., Mitchell, C., Ying, G., Potisuk, S., Srinivasan, P., Chen, R., Zoltowski, C., McPheters, L., Pellom, B., & Helzerman, R. (1994). Integrating language models with speech recognition. In *AAAI-94 Workshop on Integration of Natural Language and Speech Processing*, pp. 139–146.

Her, O.-S. (1996). Apptek machine translation toolkit user's manual. Tech. rep., Apptek Inc., McLean, VA.

Jurafsky, D., & Martin, J. (2008). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (2nd edition). Prentice Hall.

Kay, M. (1986). Algorithm schemata and data structures in syntactic processing. *Readings in natural language processing*, 35–70.

Kay, M. (1996). Chart generation. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pp. 200–204 Morristown, NJ, USA. Association for Computational Linguistics.

Kim, J.-H., & Woodland, P. C. (2001). The use of prosody in a combined system for punctuation generation. In *Proc. EUROSPEECH '01*.

Kneser, R., & Ney, H. (1995). Improved backing-off for M-gram language modeling. In *ICASSP'95: International Conference on Acoustics, Speech, and Signal Processing*, pp. 181–184 Detroit, MI, USA. IEEE Computer Society.

Köprü, S. (1999). Extendible structural transfer: A case for German to Turkish translation. Master's thesis, Computer Engineering Department, METU.

Köprü, S., Yazıcı, A., Çiloğlu, T., & Birtürk, A. (2008). Coupling speech recognition and rule-based machine translation with chart parsing. In *LangTech 2008: Proceedings of the Language and Speech Technology Conference* Rome, Italy.

Kuhn, J. (2000). Processing optimality-theoretic syntax by interleaved chart parsing and generation. In *ACL '00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pp. 368–375 Morristown, NJ, USA. Association for Computational Linguistics.

Lavie, A., & Agarwal, A. (2007). Meteor: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *ACL 2007: Proceedings of the Second Workshop on Statistical Machine Translation at the 45th Meeting of the ACL*, pp. 228–231. ACL.

Lavie, A., Levin, L., Waibel, A., Gates, D., Gavalda, M., & Mayfield, L. (1996). JANUS: a multi-lingual speech-to-speech translation system for spontaneously spoken language in a limited domain. In *AMTA'96: Proceedings of*

the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 252–255 Montreal, Quebec.

Llitjos, A. F., & Vogel, S. (2007). A walk on the other side: Adding statistical components to a transfer-based translation system. In *Proceedings of the HLT-NAACL workshop on Syntax and Structure in Statistical Translation* Rochester, NY.

Manning, C. D., & Schütze, H. (2000). *Foundations of Statistical Natural Language Processing*. The MIT Press.

Mathias, L., & Byrne, W. (2006). Statistical phrase-based speech translation. In *ICASSP'06*, Vol. 1 Toulouse, France.

Matusov, E., Kanthak, S., & Ney, H. (2005). On the integration of speech recognition and statistical machine translation. In *Interspeech*, pp. 3177–3180. ISCA Best Student Paper Award.

Merriam-Webster (2003). *Merriam-Webster's Collegiate Dictionary* (11th edition). Merriam-Webster.

Miller, G. A., & Fellbaum, C. (2008). *WordNet*. Princeton University, `http://wordnet.princeton.edu`.

Mohri, M. (1997). Finite-state transducers in language and speech processing. *Computational Linguistics*, *23*(2), 269–311.

Mohri, M., Pereira, F. C. N., & Riley, M. (1998). A rational design for a weighted finite-state transducer library. In *WIA '97: Revised Papers from the Second International Workshop on Implementing Automata*, pp. 144–158 London, UK. Springer-Verlag.

Mohri, M., & Riley, M. (1997). Weighted determinization and minimization for large vocabulary speech recognition. In *Proc. Eurospeech '97*, pp. 131–134 Rhodes, Greece.

Moore, R. C. (2004). Improved left-corner chart parsing for large context-free grammars. In Bunt, H., Carroll, J., & Satta, G. (Eds.), *New Developments in Parsing Technology*, pp. 185–201. Kluwer Academic Publishers.

Neumann, G. (1998). Interleaving natural language parsing and generation through uniform processing. *Artificial Intelligence*, *99*(1), 121–163.

Ney, H. (1991). Dynamic programming parsing for context-free grammars in continuous speech recognition. *IEEE Transactions on Signal Processing*, *39*(2), 336–340.

Ney, H. (1999). Speech translation: coupling of recognition and translation. In *ICASSP '99: Proceedings of the Acoustics, Speech, and Signal Processing, 1999. on 1999 IEEE International Conference*, pp. 517–520 Washington, DC, USA. IEEE Computer Society.

Nilsson, N. J. (1971). *Problem-Solving Methods in Artificial Intelligence*. McGraw-Hill Pub. Co.

Nirenburg, S., Carbonnell, J., Tomita, M., & Goodman, K. (1994). *Machine Translation: A Knowledge-based Approach*. Morgan Kaufmann Publishers, Los Altos, CA.

Och, F. J. (2002). *Statistical Machine Translation: From Single-Word Models to Alignment Templates*. Ph.D. thesis, RWTH Aachen University, Aachen, Germany.

Oepen, S., Velldal, E., Lonning, J. T., Meurer, P., Rosen, V., & Flickinger, D. (2007). Towards hybrid quality-oriented machine translation: On linguistics and probabilities in mt. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation*, pp. 144–153 Skovde, Sveden.

Ostendorf, M., Shafran, I., & Bates, R. (2003). Prosody models for conversational speech recognition. In *Proc. of the 2nd Plenary Meeting and Symposium on Prosody and Speech Processing*, pp. 147–154.

Paeseler, A. (1988). Modification of Earley's algorithm for speech recognition. In *Proceedings of the NATO Advanced Study Institute on Recent advances in speech understanding and dialog systems*, pp. 465–472 New York, NY, USA. Springer-Verlag New York, Inc.

Pallett, D. S., Fiscus, J. G., Fisher, W. M., Garofolo, J. S., Lund, B. A., & Przybocki, M. A. (1994). 1993 benchmark tests for the ARPA spoken language program. In *HLT '94: Proceedings of the workshop on Human Language Technology*, pp. 49–74 Morristown, NJ, USA. Association for Computational Linguistics.

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pp. 311–318 Morristown, NJ, USA. Association for Computational Linguistics.

Pentheroudakis, J., & Higginbotham, D. (1991). Morfogen: A morphology grammar builder and dictionary interface tool. Tech. rep., Brigham Young University, Provo, Utah.

Prevost, S., & Steedman, M. (1994). Specifying intonation from context for speech synthesis. *Speech Communication*, *15*, 139–153.

Quan, V. H., Federico, M., & Cettolo, M. (2005). Integrated N-best re-ranking for spoken language translation. In *Interspeech'05: Proc. Conference on Speech Communication and Technolog*, pp. 3181–3184.

Ringger, E. K. (1995). A robust loose coupling for speech recognition and natural language understanding. Tech. rep. TR592, The University of Rochester.

Roark, B. (2001). Probabilistic top-down parsing and language modeling. *Computational Linguistics*, *27*(2), 249–276.

Roark, B. (2002). Markov parsing: lattice rescoring with a statistical parser. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for*

Computational Linguistics, pp. 287–294 Morristown, NJ, USA. Association for Computational Linguistics.

Rosenkrantz, O. J., & Lewis, P. M. (1970). Deterministic left corner parsing. In *IEEE Conference Record of the 11th Annual Symposium on Switching and Automata Theory*, pp. 139–152.

Rychtyckyj, N. (2006). Machine translation for manufacturing: A case study at ford motor company. In *AAAI*.

Saleem, S., Jou, S.-C., Vogel, S., & Schultz, T. (2004). Using word lattice information for a tighter coupling in speech translation systems. In *ICSLP'04*.

Shen, W., Zens, R., Bertoldi, N., & Federico, M. (2006). The JHU Workshop 2006 IWSLT system. In *In Proc. of the International Workshop on Spoken Language Translation*, pp. 59–63.

Shieber, S. M. (1988). A uniform architecture for parsing and generation. In *Proceedings of the 12th conference on Computational linguistics*, pp. 614–619 Morristown, NJ, USA. Association for Computational Linguistics.

Shieber, S. M., van Noord, G., Moore, R. C., & Pereira, F. C. N. (1989). A semantic-head-driven generation algorithm for unification-based formalisms. In *Proceedings of the 27th annual meeting on Association for Computational Linguistics*, pp. 7–17 Morristown, NJ, USA. Association for Computational Linguistics.

Shriberg, E., & Stolcke, A. (2002). Prosody modeling for automatic speech recognition and understanding. In Ostendorf, M., Khudanpur, S., & Rosenfeld, R. (Eds.), *Proc. Workshop on Mathematical Foundations of Natural Language Modeling* Institute for Mathematics and its Applications, Minneapolis.

Shriberg, E., & Stolcke, A. (2004). Direct modeling of prosody: An overview of applications in automatic speech processing. In *Proceedings of International Conference on Speech Prosody*.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., & Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *In Proceedings of Association for Machine Translation in the Americas*, pp. 223–231.

Steedman, M. (2000). Information structure and the syntax-phonology interface. *Linguistic Inquiry*, *34*(1), 649–689.

Steel, S., & de Roeck, A. (1987). Bidirectional chart parsing. In *on Advances in Artificial Intelligence*, pp. 223–235 New York, NY, USA. John Wiley & Sons, Inc.

Stock, O., Falcone, R., & Insinnamo, P. (1988). Island parsing and bidirectional charts. In *Proceedings of the 12th conference on Computational linguistics*, pp. 636–641 Morristown, NJ, USA. Association for Computational Linguistics.

Stolcke, A. (1995). An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, *21*(2), 165–201.

Stolcke, A. (2002). SRILM - An extensible language modeling toolkit. In *Proc. International Conference on Spoken Language Processing*, Vol. 2, pp. 901–904 Denver, CO.

Sudkamp, T. A. (1991). *Languages and Machines*. Addison-Wesley.

Szaszak, G., & Vicsi, K. (2007). Speech recognition supported by prosodic information for fixed stress languages. In *Text, Speech and Dialogue*, pp. 262–269. Lecture Notes in Computer Science, Springer.

Tomita, M. (1986). An efficient word lattice parsing algorithm for continuous speech recognition. *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '86.*, *11*, 1569–1572.

Tran, B., Seide, F., & Steinbiss, V. (1996). A word graph based N-best search in continuous speech recognition. In *ICSLP'96: Proc. of the International*

Conference on Acoustics, Speech, and Signarl Processing, Vol. 4, pp. 2127–2130 Philadelphia, PA.

Tür, G., Stolcke, A., Hakkani-Tür, D., & Shriberg, E. (2001). Integrating prosodic and lexical cues for automatic topic segmentation. *Computational Linguistics*, *27*(1), 31–57.

Vidal, E. (1997). Finite-state speech-to-speech translation. In *ICASSP'97: Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 1, p. 111 Washington, DC, USA. IEEE Computer Society.

Weber, H. (1994). Time synchronous chart parsing of speech integrating unification grammars with statistics. In *Proceedings of the Eighth Twente Workshop on Language Technology*, pp. 107–119.

Weber, H., Spilker, J., & Gorz, G. (1997). Parsing N best trees from a word lattice. In *KI - Kunstliche Intelligenz*, pp. 279–288.

Wessel, F., Schluter, R., Macherey, K., & Ney, H. (2001). Confidence measures for large vocabulary continuous speech recognition. *IEEE Transactions on Speech and Audio Processing*, *9*(3), 288–298.

Woodland, P. (2000). *HTK Speech Recognition Toolkit*. Cambridge University Engineering Department, `http://htk.eng.cam.ac.uk`.

Xu, P., Chelba, C., & Jelinek, F. (2002). A study on richer syntactic dependencies for structured language modeling. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pp. 191–198. Association for Computational Linguistics.

Younger, D. H. (1967). Recognition and parsing of context-free languages in time n^3. *Information and Control*, *10*, 189–208.

Zajac, R. (1998). Feature structures, unification and finite-state transducers. In *FSMNLP'98: International Workshop, on Finite State Methods in Natural Language Processing.*

Zhang, R., & Kikui, G. (2006). Integration of speech recognition and machine translation: Speech recognition word lattice translation. *Speech Communication*, *48*(3-4), 321–334.

APPENDIX A

WORD GRAPH PRUNING SCRIPT

```
:::: clean up previous files
del *fsm
del *pdf

for %%1 in (*.slf) do (
:::: Convert slf format to at&t format
slf2fsm %%~n1.slf

:::: Compile node-labeled lattice into an fsm as a reference
rem fsmcompile -s%%~n1.sta -F%%~n1.lat.fsm %%~n1.lat
rem fsmdraw -p -s%%~n1.sta %%~n1.lat.fsm
    | dot -Tpdf > %%~n1.lat.fsm.pdf
rem del %%~n1.lat.fsm
del %%~n1.lat
del %%~n1.sta

:::: Compile the arc-labeled fsm
fsmcompile -i%%~n1.sym -F%%~n1.fsm %%~n1.txt
rem fsmdraw -p -i%%~n1.sym %%~n1.fsm
    | dot -Tpdf > %%~n1.fsm.pdf
del %%~n1.txt

:::: nfa 2 dfa conversion of the fsm
fsmdeterminize -F%%~n1.det.fsm %%~n1.fsm
rem fsmdraw -p -i%%~n1.sym %%~n1.det.fsm
| dot -Tpdf > %%~n1.det.fsm.pdf
del %%~n1.fsm

:::: Remove epsilons
fsmrmepsilon -F%%~n1.noeps.det.fsm %%~n1.det.fsm
del %%~n1.det.fsm

:::: Minimize the deterministic fsm
```

```
fsmminimize -F%%~n1.mini.fsm %%~n1.noeps.det.fsm
rem fsmdraw -p -i%%~n1.sym %%~n1.mini.fsm
    | dot -Tpdf > %%~n1.mini.fsm.pdf
del %%~n1.noeps.det.fsm

:::: Extract First-best hypothesis
fsmbestpath -F%%~n1.first1.fsm %%~n1.mini.fsm
rem fsmdraw -p -i%%~n1.sym %%~n1.first1.fsm
    | dot -Tpdf > %%~n1.first1.fsm.pdf
fsmprint -i%%~n1.sym %%~n1.first1.fsm > %%~n1.first1.fsm.txt

:::: Extract 10-best hypotheses
fsmbestpath -n 10 -F%%~n1.first10.fsm %%~n1.mini.fsm
fsmdeterminize -F%%~n1.first10.det.fsm %%~n1.first10.fsm
fsmrmepsilon -F%%~n1.first10.noeps.fsm %%~n1.first10.det.fsm
fsmminimize -F%%~n1.first10.fsm %%~n1.first10.noeps.fsm
rem fsmdraw -p -i%%~n1.sym %%~n1.first10.fsm
    | dot -Tpdf > %%~n1.first10.fsm.pdf
fsmprint -i%%~n1.sym %%~n1.first10.fsm > %%~n1.first10.fsm.txt
del %%~n1.first10.det.fsm
del %%~n1.first10.noeps.fsm

:::: Extract 30-best hypotheses
fsmbestpath -n 30 -F%%~n1.first30.fsm %%~n1.mini.fsm
fsmdeterminize -F%%~n1.first30.det.fsm %%~n1.first30.fsm
fsmrmepsilon -F%%~n1.first30.noeps.fsm %%~n1.first30.det.fsm
fsmminimize -F%%~n1.first30.fsm %%~n1.first30.noeps.fsm
rem fsmdraw -p -i%%~n1.sym %%~n1.first30.fsm
    | dot -Tpdf > %%~n1.first30.fsm.pdf
fsmprint -i%%~n1.sym %%~n1.first30.fsm > %%~n1.first30.fsm.txt
del %%~n1.first30.det.fsm
del %%~n1.first30.noeps.fsm

:::: Extract 50-best hypotheses
fsmbestpath -n 50 -F%%~n1.first50.fsm %%~n1.mini.fsm
fsmdeterminize -F%%~n1.first50.det.fsm %%~n1.first50.fsm
fsmrmepsilon -F%%~n1.first50.noeps.fsm %%~n1.first50.det.fsm
fsmminimize -F%%~n1.first50.fsm %%~n1.first50.noeps.fsm
rem fsmdraw -p -i%%~n1.sym %%~n1.first50.fsm
    | dot -Tpdf > %%~n1.first50.fsm.pdf
fsmprint -i%%~n1.sym %%~n1.first50.fsm > %%~n1.first50.fsm.txt
del %%~n1.first50.det.fsm
del %%~n1.first50.noeps.fsm

:::: Extract 100-best hypotheses
fsmbestpath -n 100 -F%%~n1.first100.fsm %%~n1.mini.fsm
```

```
fsmdeterminize -F%%~n1.first100.det.fsm %%~n1.first100.fsm
fsmrmepsilon -F%%~n1.first100.noeps.fsm %%~n1.first100.det.fsm
fsmminimize -F%%~n1.first100.fsm %%~n1.first100.noeps.fsm
rem fsmdraw -p -i%%~n1.sym %%~n1.first100.fsm
    | dot -Tpdf > %%~n1.first100.fsm.pdf
fsmprint -i%%~n1.sym %%~n1.first100.fsm > %%~n1.first100.fsm.txt
del %%~n1.first100.det.fsm
del %%~n1.first100.noeps.fsm
del %%~n1.mini.fsm
rem del %%~n1.sym
)
```

APPENDIX B

SAMPLE WORD GRAPH

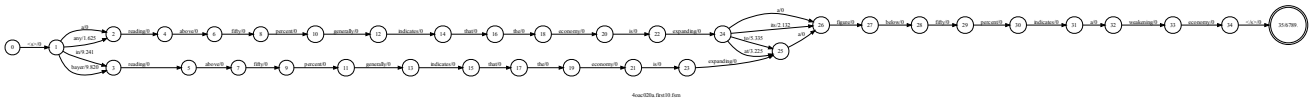

Figure B.1: Pruned lattice 4oac020a from the NIST HUB1 data set containing 10 hypotheses.

Corresponding listing for the word graph in Figure B.1.

```
0       1       <s>
1       2       a
1       2       any     1.62598
1       3       in      9.24121
1       3       bayer   9.82031
2       4       reading
3       5       reading
4       6       above
5       7       above
6       8       fifty
7       9       fifty
8       10      percent
9       11      percent
10      12      generally
11      13      generally
12      14      indicates
```

```
13      15      indicates
14      16      that
15      17      that
16      18      the
17      19      the
18      20      economy
19      21      economy
20      22      is
21      23      is
22      24      expanding
23      25      expanding
24      26      a
24      25      to      5.33594
24      26      its     2.13281
24      25      at      3.22559
25      26      a
26      27      figure
27      28      below
28      29      fifty
29      30      percent
30      31      indicates
31      32      a
32      33      weakening
33      34      economy
34      35      </s>
35      6789.37
```

APPENDIX C

SAMPLE C-STRUCTURE

```
TILT:1139
 | BOSS:1
 | NP:351
 |  | NP3:315
 |  |  | QFR:2 a
 |  |  | NP2:272
 |  |  |  | NP1:249
 |  |  |  |  | N:70
 |  |  |  |  |  | NSTEM:8 reading
 |  |  |  |  | PP:223
 |  |  |  |  |  | PBAR:191
 |  |  |  |  |  |  | P:13 above
 |  |  |  |  |  |  | NP:167
 |  |  |  |  |  |  |  | NP3:159
 |  |  |  |  |  |  |  |  | NP2:156
 |  |  |  |  |  |  |  |  |  | NP1:149
 |  |  |  |  |  |  |  |  |  |  | N:137
 |  |  |  |  |  |  |  |  |  |  |  | QFR:101
 |  |  |  |  |  |  |  |  |  |  |  |  | QNUM:76
 |  |  |  |  |  |  |  |  |  |  |  |  |  | QSTEM:18 fifty
 |  |  |  |  |  |  |  |  |  |  |  | PERCENT:20 percent
 | ADVP:169
 |  | ADVBAR:125
 |  |  | ADV|L:103
 |  |  |  | ADV:78
 |  |  |  |  | ADVSTEM:22 generally
 | V|R:854
 |  | V:25 indicates
 |  | NP:848
 |  |  | NP3:794
 |  |  |  | DET:30 that
 | NP:209
 |  | NP3:151
```

```
|  |  | DET:34 the
|  |  | NP2:129
|  |  |  | NP1:107
|  |  |  |  | N:82
|  |  |  |  |  | NSTEM:36 economy
| V|R:452
|  | V:38 is
| V|R:527
|  | V:44 expanding
|  | PP:394
|  |  | PBAR:380
|  |  |  | P:53 at
|  |  |  | NP:364
|  |  |  |  | NP3:328
|  |  |  |  |  | QFR:54 a
|  |  |  |  |  | NP2:283
|  |  |  |  |  |  | NP1:264
|  |  |  |  |  |  |  |  | N:87
|  |  |  |  |  |  |  |  |  | NSTEM:56 figure
|  |  |  |  |  |  |  |  | PP:237
|  |  |  |  |  |  |  |  |  | PBAR:200
|  |  |  |  |  |  |  |  |  |  | P:59 below
|  |  |  |  |  |  |  |  |  |  | NP:176
|  |  |  |  |  |  |  |  |  |  |  | NP3:164
|  |  |  |  |  |  |  |  |  |  |  |  | NP2:163
|  |  |  |  |  |  |  |  |  |  |  |  |  | NP1:162
|  |  |  |  |  |  |  |  |  |  |  |  |  |  | NP1:155
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | N:145
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | QFR:114
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | QNUM:90
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | QSTEM:60 fifty
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | PERCENT:61 percent
```

APPENDIX D

SAMPLE F-STRUCTURE

```
C 527, 7689, (12-22), V|R, VBARn 3  w:-22,  b

 | [
 | RULES [ 1 [ V|R $ PREVERBAL, V|RCOMMA $ ] ]
 | TNOD
 | | [
 | | VFORM ING
 | | NEEDS
 | | | [
 | | | 27 [ PFORM at ]
 | | | C-OB  {
 | | | | | [
 | | | | | PFORM on
 | | | | | STRINGS [ STRING2 on sth ]
 | | | | | ],
 | | | | | [
 | | | | | PFORM into
 | | | | | STRINGS [ STRING2 into sth ]
 | | | | | ] }
 | | | C-ACOB  {
 | | | | | [
 | | | | | PFORM into
 | | | | | STRINGS [ STRING2 sth into sth ]
 | | | | | ] }
 | | | ]
 | | AMBIG MINUS
 | | ALLOWS [ C-OBJ1 $ STD-FE-HATE-TIME-OBJ $ ]
 | | HATES [
 | | | C-INTRDEST $ STD-FE-HATE-OB $
 | | | C-INTRLOC $ STD-FE-HATE-OB $
 | | | C-INTR $ STD-FE-HATE-OB $
 | | | C-OBJ1 $ STD-FE-HATE-ACOB, STD-FE-HATE-ACC-PPOB $
 | | | C-ACCLOC $ STD-FE-HATE-ACOB $
```

```
| | | C-ACCDEST $ STD-FE-HATE-ACOB $
| | | ]
| | ]
| FE
| | [
| | 0 [ CTYPE-N OPT VBAR OPT ]
| | 1 [
| | | VBAR $ STD-FE-ACC, STD-FE-INTR, STD-FE-OB, STD-FE-ACOB $
| | | AUX $ STD-FE-NOAUX $
| | | CTYPE-N $ STD-FE-ACC, STD-FE-INTR, STD-FE-OB, STD-FE-ACOB $
| | | CTYPE-AJL-PRENP $ STD-FE-ACC-AJL, STD-FE-INTR-AJL, STD-FE-ACOB-AJL $
| | | WHWORD $ STD-FE-ACC-REL, FAIL-FE, STD-FE-ACOB-REL $
| | | ]
| | ]
| FS
| | [
| | CTOPIC MINUS
| | GAP MINUS
| | SUBJ
| | | [
| | | DEAR MINUS
| | | REEFER PLUS
| | | ]
| | COMPID 3001
| | PRED # SUBJ, OBJ1 #
| | PTOPIC MINUS
| | FORM expand
| | CTYPE-N $ STD-COMPL-ACC, STD-COMPL-INTR, STD-COMPL-OB, STD-COMPL-ACOB $
| | VERBAL PLUS
| | NOMINAL MINUS
| | WALANG-TYPE WV
| | CANBE # WV, WN, WADJ, WING, WEN #
| | VOICE ACTIVE
| | PROGRES PLUS
| | CAPFLAG MINUS
| | ACTFORM expanding
| | ADJUNCTS  {
| | | | [
| | | | PFORM at
| | | | PSLOT SLONP
| | | | PCASE # LOCT, TEMP #
| | | | WALANG-RES P
| | | | DEFINITE MINUS
| | | | WHATAMI MINUS
| | | | COUNT PLUS
| | | | ADJUNCTS  {
```

```
| | | | | | [
| | | | | | PFORM below
| | | | | | PSLOT # SLOVP, SLONP, SLADJ #
| | | | | | PCASE LOCT
| | | | | | WALANG-RES P
| | | | | | LONGNOSE PLUS
| | | | | | PERSON THIRD
| | | | | | DEFINITE MINUS
| | | | | | FORM indicates
| | | | | | PROFORM IT
| | | | | | USE GENERAL
| | | | | | NUMBER PL
| | | | | | VERBAL MINUS
| | | | | | NOMINAL PLUS
| | | | | | REEFER PLUS
| | | | | | WALANG-TYPE WN
| | | | | | ENDING other
| | | | | | NOMAD
| | | | | | | [
| | | | | | | CASE MINUS
| | | | | | | DEFINITE UNMARKED
| | | | | | | ENDING other
| | | | | | | REEFER PLUS
| | | | | | | WALANG-TYPE # WNUM, WN #
| | | | | | | VERBAL MINUS
| | | | | | | NOMINAL PLUS
| | | | | | | PROPER PLUS
| | | | | | | COUNT PLUS
| | | | | | | NOMAD MINUS
| | | | | | | PROFORM IT
| | | | | | | USE # PART, MONEY, MEASURE #
| | | | | | | FORM %
| | | | | | | QUANTIFIER  {
| | | | | | | | | [
| | | | | | | | | CARD PLUS
| | | | | | | | | ACTFORM ten
| | | | | | | | | QFORM NUMERIC
| | | | | | | | | CAPFLAG MINUS
| | | | | | | | | ABBFORM fifty
| | | | | | | | | MULTIPLIER
| | | | | | | | | | [
| | | | | | | | | | FORM five
| | | | | | | | | | FINGER 5
| | | | | | | | | | ]
| | | | | | | | | POWER TEN1
| | | | | | | | | COUNT PLUS
```

```
| | | | | | | | | WALANG-TYPE WNUM
| | | | | | | | | DEGREE MINUS
| | | | | | | | | FORM ten
| | | | | | | | | ] }
| | | | | | | | ]
| | | | | | | ] }
| | | | ACTFORM figure
| | | | KYUEKS MINUS
| | | | KYU MINUS
| | | | EKS MINUS
| | | | ESS MINUS
| | | | NUMBER SG
| | | | USE GENERAL
| | | | CAPFLAG MINUS
| | | | PROFORM IT
| | | | HEWMN MINUS
| | | | ANIM MINUS
| | | | PROPER MINUS
| | | | PERSON THIRD
| | | | PASSITEM MINUS
| | | | MROFBBA MINUS
| | | | ABBFORM MINUS
| | | | REEFER PLUS
| | | | WALANG-TYPE WN
| | | | NOMINAL PLUS
| | | | VERBAL MINUS
| | | | FORM figure
| | | | APPLE MINUS
| | | | GAP OPT
| | | | ENDING other
| | | | QUANTIFIER  {
| | | | | | [
| | | | | | CARD PLUS
| | | | | | DEGREE MINUS
| | | | | | WALANG-TYPE WNUM
| | | | | | COUNT PLUS
| | | | | | QFORM NUMERIC
| | | | | | DEFINITE MINUS
| | | | | | NUMBER SG
| | | | | | SARPFORM ANY
| | | | | | POWER TENO
| | | | | | FINGER 1
| | | | | | FORM a
| | | | | | ] }
| | | | ] }
| | OBJ1
```

```
| | | [
| | | CASE ACC
| | | PERSON THIRD
| | | DEFINITE MINUS
| | | FORM weaken
| | | VERBAL PLUS
| | | NOMINAL MINUS
| | | WALANG-TYPE WV
| | | CANBE # WV, WN, WADJ, WING, WEN #
| | | CTYPE-N $ STD-COMPL-INTR $
| | | VOICE ACTIVE
| | | PROGRES PLUS
| | | CAPFLAG MINUS
| | | ACTFORM weakening
| | | PROFORM IT
| | | NUMBER SG
| | | USE ING
| | | PROPER MINUS
| | | PRIORESS 1
| | | PRIOR 1
| | | COUNT MINUS
| | | HEWMN MINUS
| | | ANIM MINUS
| | | REEFER PLUS
| | | ENDING cn
| | | QUANTIFIER  {
| | | | | [
| | | | | CARD PLUS
| | | | | DEGREE MINUS
| | | | | WALANG-TYPE WNUM
| | | | | COUNT PLUS
| | | | | QFORM NUMERIC
| | | | | DEFINITE MINUS
| | | | | NUMBER SG
| | | | | SARPFORM ANY
| | | | | POWER TENO
| | | | | FINGER 1
| | | | | FORM a
| | | | | ] }
| | | ]
| | ]
| CAT V|R
| SPRE MINUS
| ]
```

VITA

Selçuk Köprü was born in Lemgo, Germany, on May 29, 1973. In 1996, he received his B. Sc. degree from the Department of Computer Engineering, Middle East Technical University. Following his graduation, he joined the M. Sc. program in the same department where he also worked as a teaching assistant between 1998 and 2000. His masters thesis was about machine translation from German to Turkish. After the graduation in 1999, he was accepted to the Ph. D. program in the same department. In 2001, he set up Teknoloji Yazılımevi in METU Technopolis. He has been also working for Apptek since 1999 as a senior computational linguist. He took part in the development of industry level MT systems between many language pairs including English, Turkish, Arabic, Indonesian and Pashto. His research interests include machine translation and speech recognition.