

# ROBUST PARSING FOR WORD LATTICES IN CONTINUOUS SPEECH RECOGNITION SYSTEMS

S. Momtazi, H. Sameti, M. Fazel-Zarandi, M. Bahrani  
Speech Processing Lab, Computer Engineering Department,  
Sharif University of Technology, Tehran, Iran  
momtazi,fazel,bahrani@ce.sharif.edu, sameti@sharif.edu

## ABSTRACT

*One of the roles of a Natural Language Processing (NLP) model in Continuous Speech Recognition (CSR) systems is to find the best sentence hypothesis by ranking all  $n$ -best sentences according to the grammar. This paper describes a robust parsing algorithm for Spoken Language Recognition (SLR) which utilizes a technique that improves the efficiency of parsing. This technique integrates grammatical and statistical approaches, and by using a best-first parsing strategy improves the accuracy of recognition. Preliminary experimental results using a Persian continuous speech recognition system show effective improvements in accuracy with little change in recognition time. The word error rate was also reduced by 18%.*

## 1. INTRODUCTION

Parsing speech is a difficult task since the input for the parsing system is the output of some speech recognition application which may include some errors due to noises, false starts, self repairs and ungrammatical constructions that are produced by the speakers or the recognition system. As a result no grammatical sentence can be found for some of these inputs, and hence the purpose of a natural language processing unit in these cases would be to find the best grammatical structure representing the input. To reach this, the best way is the construction of a robust parsing algorithm which can even handle ill-formed sentences [1]. There are two approaches for robust parsing: 1) error correction, and 2) partial parsing [6]. The approach inherited by this system is based on partial parsing and overcomes this problem by introducing some criteria which will be discussed in the following sections.

The organization of this paper is as follows: in section 2 the concept of word lattice, what it means and how it is constructed, is introduced. Section 3 deals with grammatical and statistical models. The partial robust parsing algorithm and its preliminary experimental results are discussed in sections 4 and 5. Finally, in section 6,

concluding remarks are made.

## 2. WORD LATTICE

The input for the system can be in two forms:  $n$ -best list (Fig. 1), or  $n$ -best lattice (Fig. 2). In the former, all first  $n$  sentences are parsed separately, whereas in the latter, all these sentences are merged and the parse algorithm operates on them simultaneously. This characteristic of  $n$ -best lattice has major effects on performance [5], because the words which are repeated in many hypotheses are only processed once. To make use of this, at the beginning the system converts the output of the continuous speech recognition system - which is in the form of  $n$ -best list - into  $n$ -best lattice form, called word lattice.

A word lattice consists of a set of word hypotheses produced by continuous speech recognition systems. Each word hypotheses has a beginning and an ending point, denoted by  $begin(w)$  and  $end(w)$ , which are directly related to the time it was recognized by the acoustic signal processor [4]. The word lattice used in this system is implemented as a data structure in which every word is linked to the words that immediately follow it in different sentences.

One of the major problems in spoken language processing is the existence of noise in the environment that the algorithm must be able to handle. Yan, Zheng, and Xu [8] suggested the extension of the grammar by introducing new by-passing grammar rules. Since the purpose of this system is to present a model without changing the existing grammar, another solution was devised that is implemented in the word lattice. This solution only considers part of these rules at present, but is extendable. The remaining rules that should be extended in the future are to by-pass words that none of the sentence hypotheses detect as noise. In order to implement by-passing rules in the word lattice, addition to the words coming exactly after each other in  $n$ -best list, every word  $w_a$  is linked to any other word  $w_b$  satisfying the conditions: 1)  $end(w_a) < begin(w_b)$ ; and 2) there does not exist any  $w_c$  such that  $end(w_a) < begin(w_c)$  and  $end(w_c) < begin(w_b)$ , (the dashed line in Fig. 2).



Figure 1. The n-best list input (input: "Nobody likes lies.")

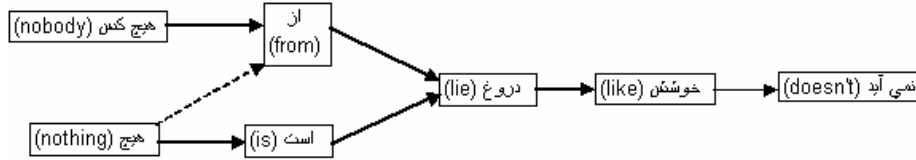


Figure 2. The n-best lattice input (input: "Nobody likes lies."). The dashed line represents a jump edge.

### 3. GRAMMATICAL AND STATISTICAL MODEL

In this section, the language processing model comprised of grammatical and statistical parts is discussed.

#### 3.1. The Grammatical Model

The grammar used here includes a lexicon and a set of extended rules. The lexicon contains features for every word and all possible part of speeches (POS) for it, and the extended rules are employed to describe how larger constituents can be made and how the features are related to each other.

The main disadvantage of these grammars is the inability to parse ill-formed sentences. Another disadvantage is that because of high degree of ambiguity, very often more than one grammatically valid sentence may be produced. No criteria to distinguish between these sentences in the syntactic level exist. One way to handle the preceding problems is to integrate the grammatical approach with statistical approaches.

#### 3.2. The Statistical Model

To deal with ambiguity in grammar, statistical models are used to find the priority of produced sentences. The general method to reach this goal is to use some of the local context of the sentence in which the word appears [1]. In this case the probability for a word to be used in a sentence should be conditioned upon the words previously used.

Let  $w_1, \dots, w_T$  be a sentence hypothesis denoted by  $S$ . The probability  $P(S)$  is calculated as follow:

$$P(S) = \prod_{i=1}^T P(w_i | w_{i-1}, w_{i-2}, \dots, w_1) \quad (1)$$

There are still no effective methods for calculating the probability of these long sequences accurately, as it would require far too much data [1]. In this research the bi-gram model is assumed:

$$P(w_i | w_{i-1}, w_{i-2}, \dots, w_1) \cong P(w_i | w_{i-1}) \quad (2)$$

and these probabilities are retrieved from a large database of training corpus.

In order to integrate this into the parsing algorithm, each constituent and arc is assigned a probability. When extending an arc with a constituent, the probability for the arc is updated using the following formula:

$$P(na) = P(oa) \times P(c) \times P(w_{c1} | w_{oat}) \quad (3)$$

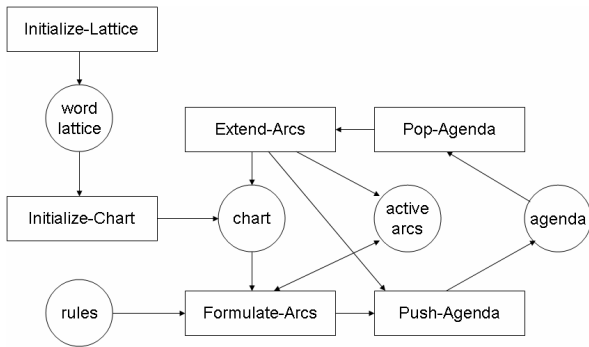
where  $na$ ,  $oa$ , and  $c$  stand for new-arc, old-arc, and constituent respectively, and  $w_{c1}$  and  $w_{oat}$  denote the starting word of the constituent and the ending word of the last constituent of old-arc from which it is extended. This probability can be easily calculated in each parsing step. The bi-gram probability can be calculated between two words or two POSs, but the precision is higher using word bi-gram probabilities.

### 4. PARTIAL ROBUST PARSING ALGORITHM

The parsing algorithm consists of five global data structures: *rules*, *word-lattice*, *chart*, *active-arcs*, *agenda*; and one main procedure called *Parse* with six assistant procedures: *Initialize-Lattice*, *Initialize-Chart*, *Formulate-Arcs*, *Push-Agenda*, *Pop-Agenda*, and *Extend-Arcs*. An abstract diagram shown in Fig. 3 indicates the relationships between the procedures and the global data structures.

*Rules* is a collection of all the grammar rules used for parsing, which can also have features. For keeping the

input, a *word lattice* is constructed according to the details given in section 2. The main data structure is the *chart* which is somewhat different from the traditional model used in chart parsing. The conventional model is used for parsing known sentences with fixed start and end points, whereas the augmented chart is constructed based on the word lattice and is filled at first with all POSs derived from it. The chart is in the form of a directed-acyclic graph (DAG) in which every completed constituent is inserted by pointing to its sub-constituents. *Active-arcs* contain all the rules activated by the algorithm. The *agenda* is implemented as a priority queue because the parsing algorithm is best-first. This data structure contains all arcs ready for extension using a certain constituent. The queue is arranged by the predicted probability for each arc-constituent pair – the probability of extending an arc with a constituent as explained in section 3.2.



**Figure 3.** The relations among data structures and assistant procedures

Initially, the input is passed to the parser, which call *Initialize-Lattice*, to convert the n-best list input to n-best lattice and create the word lattice used to hold the input in this form. Then, all POSs of the input words and their probabilities are added to the chart using procedure *Initialize-Chart*. After these two steps, the parser calls *Formulate-Arcs* on all POSs. This procedure performs two actions. First, it adds all rules starting with a certain constituent to active-arcs. Second, it adds all active-arcs extendable using this constituent to agenda with a call to *Push-Agenda*. The input to *Push-Agenda* is an active-arc and a constituent. This procedure calculates the probability of extending the arc using the constituent and pushes this triple set of information to the agenda.

The triple having the highest probability is then popped using procedure *Pop-Agenda* and passed to *Extend-Arcs*. If the arc is extendable by using the constituent and unification is successful, then a new arc or constituent is created by this extension. In the case of reaching the end of an arc, a constituent must be created, its probability set, added to the *chart*, and *Formulate-Arcs* called on it. Otherwise, a new extended active-arc must be created, its probability updated, and added to *active-arcs*. Since the algorithm is best-first, there is a chance that the next

constituent needed by this new active-arc was created and added to the chart earlier; therefore, the *chart* must be searched for that constituent and if any constituent is found, it must be pushed into the *agenda*. This continues until, either the threshold number of grammatical sentences with full coverage are found or that the *agenda* is empty. According to Chein, Chen, and Lee [3] the threshold suggested is three, but in this system the threshold is set to five. If the *agenda* is empty and no satisfactory sentence has been found, some criteria must be taken into account for partial parsing, and a ranking of high level constituents according to these criteria must be given.

The criteria introduced for this reason are coverage [7] and probability [3]. The coverage is calculated using begin and end points for each constituent. The constituents having the highest coverage are chosen, and among constituents with equal coverage, the one having the higher probability is ranked first. If this best constituent has full coverage, the system returns it as its result; otherwise the first sentence hypothesis among n-best sentences that contains this constituent is selected as the result. This can be further extended by ranking all sentences containing this constituent according to the probabilities of the rest of the sentence. This can be implemented by searching the chart to find constituents that do not overlap with the main constituent and are also part of those sentences.

## 5. EVALUATION

A prototype of this system has been implemented and tested on a lexicon of 1092 words, a bi-gram language model, and a set of *Persian* unification grammar rules with 25 different POSs. The speech recognition system used is SHARIF [2], which is a Persian speaker independent CSR system that uses Hidden Markov Models trained by FARSDAT speech database, to perform mono-phone modeling. The FARSDAT contains 6080 different sentences which exhibited in 5 hours and 5940 sentences of them used for training phase and the 140 remaining sentences used as testing data. All of these sentences are both grammatically correct and incorrect. The baseline results are poor because the training data is lake and mono-phones are used. The system was tested with 5-best, 7-best, and 10-best sentence hypothesizes as input to the parser. These numbers were chosen with the knowledge that on average the best sentence hypothesis is seen among the six first sentences. The results are shown in Table 1.

**Table 1.** Word error rates before applying the robust parsing algorithm (baseline) and after applying it using 5-best, 7-best, and 10-best sentence hypothesizes

Baseline	5-best	7-best	10-best
43.89%	36.79%	35.74%	36.19%

According to the results obtained, the input of the system should be limited to the 7-best sentence hypothesizes. This number is selected due to a trade-off between accuracy and recognition time. With this input, the word error rate reduction is 18.6% (it is absolutely reduced by 8.15%).

## 6. CONCLUDING REMARKS

This paper introduced a robust parsing algorithm for spoken language recognition which uses grammatical and statistical approaches to improve the efficiency of word lattice parsing. The word lattice is constructed uniquely by including special jump edges for handling noises. Because of the importance of probability in dealing with ambiguity in grammar, a best-first strategy is used. The criteria for robustness implemented in this system are coverage and probability.

Although the algorithm was tested using a Persian grammar, the underlying algorithm is language independent. Solutions to improve the results include improving the grammar, extending the POSs, and modifying n-gram probabilities.

## 7. ACKNOWLEDGEMENT

The research reported here was supported by Asr-Gooyesh research center in affiliation with Sharif University of Technology, as part of the "Continuous Speech Recognition Project". The authors are grateful to Niloofar Montazeri of Sharif University of Technology, for her advice.

## REFERENCES

- [1] Allen, J., *Natural Language Understanding*. The Benjamin-Cummings Publishing Company, Inc., 1995.
- [2] Babaali, B., Sameti, H., "The Sharif Speaker-Independent Large Vocabulary Speech Recognition System", in *Proc. 2<sup>nd</sup> Workshop on Information Technology & Its Disciplines*, Kish Island, Iran, 2004.
- [3] Chein, L. F., Chen, K. J., and Lee, L. S., "A Best-First Language Processing Model Integrating the Unification Grammar and Markov Language Model for Speech Recognition Applications". *IEEE Trans on Speech and Audio Processing*, vol.1, no. 2, 1993.
- [4] Chein, L. F., Chen, K. J., and Lee, L. S., "An Augmented Chart Data Structure with Efficient Word Lattice Parsing Scheme In Speech Recognition Application". in *Proc. 13<sup>th</sup> International Conference on Computational Linguistics*, pp. 60-65, 1990.
- [5] Collins, C., Carpenter, B., and Penn, G., "Head-Driven Parsing for Word Lattices". in *Proc. 2004 Meeting of the Association for Computational Linguistics*, 2004.
- [6] Rupp, C. J., and Milward, D., *A Robust Linguistic Processing Architecture*, Ph.D. Thesis, SIRIDUS Deliverable 4.1, Gothenburg University, Gothenburg, Sweden, 2000.
- [7] Wang, Y. Y., "A Robust Parser for Spoken Language Understanding". in *Proc. 6th European conference on speech communication and technology*, pp. 2055-2058, 1999.
- [8] Yan, P., et al, "Robust Parsing in Spoken Dialogue Systems". in *Proc. 7<sup>th</sup> European Conference on Speech Communication and Technology*, pp. 2149-2152, 2001.