BTH: An Efficient Parsing Algorithm for Word-Spotting

Yasuyuki KONO, Takehide YANO, and Munehiko SASAJIMA

Kansai Research Laboratories, Toshiba Corporation
8-6-26 Motoyama-Minami, Higashi-Nada, Kobe 658-0015, Japan
{kono,yano,sasa}@krl.toshiba.co.jp

ABSTRACT

This paper presents a new parsing algorithm, BTH, which is capable of efficiently parsing a keyword lattice that contains a large number of false alarms. The BTH parser runs without unfolding the given keyword lattice, and thus it can efficiently obtain a set of word sequences acceptable to the given grammar as the parser result.

The algorithm has been implemented on Windows-based PCs and is tested by applying it to a car navigation task that has a scale of practical applications. The result indicates promise in implementing the function of a sentence utterance-based spontaneous speech understanding in next-generation car navigation systems.

1. INTRODUCTION

Since spoken language is not only an essential means of communication but also the most natural one for human beings. Users desire computer systems with this type of communication. We had previously developed TOSBURG-II, a spoken language dialogue system [1][2]. It picked up the acceptable keyword sequence as a sentence from the keyword lattice obtained from a word-spotting engine for the task of selling hamburgers with a very limited vocabulary, and performed the semantic analysis / intention understanding. The system recognized only meaningful words, i.e., nouns and verbs, received a plausible keyword sequence as a sentence, and responded to the user by synthesized voice. The TOSBURG-II study revealed the followings: (1) by applying word-spotting technology, it is possible to cope with phenomena characteristics of spoken language such as unnecessary words or unfixed forms, (2) false alarms which are unavoidable in the word-spotting process can be also corrected by sentence analysis.

In order to expand the scale of the task which a spoken language dialogue system is capable of executing and to apply the system for practical tasks, it is necessary to correspond to a large vocabulary and a large-scale grammar. However, expansion of vocabulary and grammar causes the number of recognized words and the size of the corresponding keyword lattice to increase explosively. This poses a problem in that the processing time required for analysis becomes huge.

We have developed BTH, Bun (meaning "sentence" in Japanese) Template Hash, a parsing algorithm which is capable of efficiently parsing a keyword lattice with a large number of false alarms. When a keyword lattice is large, the list of possible word sequence generated by unfolding the lattice is huge too. Parsing the candidates in the list one by one causes an explosion of calculation time. The BTH parser runs without unfolding the given keyword lattice. This parser (1) holds hash tables for each sentence-type template, (2) calculates the set of sentence-types to which each node in the lattice can belong by referring to the table, and (3) propagates the list to the nodes following each node. It can efficiently obtain a set of acceptable word sequences to the given grammar as the parser result.

We have implemented this algorithm on PCs and tested the method by applying it to a car navigation task. In this work, the vocabulary for recognition exceeds 700 words, and the keyword lattice that can generate over 1,000,000 candidate word sequences is usually obtained as a recognition result. Also, parsing of such a large-scale lattice can usually be completed in real time in around 1 second by a PC of some generations ago (486DX2 50MHz), i.e., about 35 MIPS. The result shows promise in implementing the function of spoken-language understanding from sentence utterance in next-generation car navigation systems equipped with RISC MPUs of about 70 MIPS.

2. THE BTH PARSER

Our research goal is to develop a generic framework for spoken interfaces that runs on practical applications with considerable scale of task. We cannot expect a 100% recognition rate with current speech recognition technologies, even if strong constraints are applied as a language model. It is especially difficult to correctly recognize bound words, i.e., words that have no meanings themselves, e.g., postpositions. We have designed our speech understanding mechanism based on the following policies:

- Embodying scalability so that it works well in practical applications.
- Analyzing/understanding a user's spontaneous utterance with the grammar that accepts a sentence as a word-sequence consisting only of free words, i.e., words that have meanings themselves, e.g., nouns and verbs.
- Recognizing a user's utterance by the keyword-spotting technique. A list of possible word sequences is obtained by parsing a keyword-lattice generated from a word-spotting result.

To realize our goal, it is crucial to establish the technology for post-recognition process that can efficiently extract plausible sentences/word-sequences by dealing with the keyword lattice obtained from the word-spotting engine. We have developed the BTH parser which is capable of efficiently parsing a keyword lattice that contains a large number of false alarms. This section describes its features and techniques.

2.1. Keyword-Lattice Parsing Problem

One technique for parsing a lattice structure that is obtained by speech/handwriting recognition is to sequentially apply a natural language parser one by one to candidate sentences which are generated by unfolding the lattice. The TOSBURG-II employed a generalized LR parser [3], which analyzes a keyword lattice generating midway candidates and pruning hopeless paths [1,2]. The keyword lattice, however, generated by a speech recognizer with a large volume of vocabulary and grammar, contains complicated structure with many nodes and forms in cases. It is hard to analyze each candidate generated by unfolding the lattice in the light of both calculation time and memory.

It is essential to retrieve huge amount of both vocabulary and grammar to develop a speech interface of a practical application. We collected 389 example sentence utterances that are natural and needed by drivers who use car navigation systems with speech recognition. They include actual utterances made by one of us while driving a car and while being a passenger in a car. Then we analyzed them and made grammatical rules that apply to those examples. We also designed structures of user's intentions by classifying the collected utterances into 5W (Where, What, Which, When, Why) and 2H (How, How-Much).

In parallel with the analysis of sentence patterns and grammar, we picked up keywords from the example utterances. For instance, the sentence "Deguchi-no-mae-no-saigo-no-service-area-ha-doko? (Where is the last service area before the exit?)" was classified into the sentence-type group of "Where", and five words, i.e., deguchi (exit), mae (before), saigo (the last), service-area, and doko (where), were selected as keywords. As a result of the analysis, about 700 words were picked up as recognized keywords and were classified into about 110 word-classes (parts of speeches).

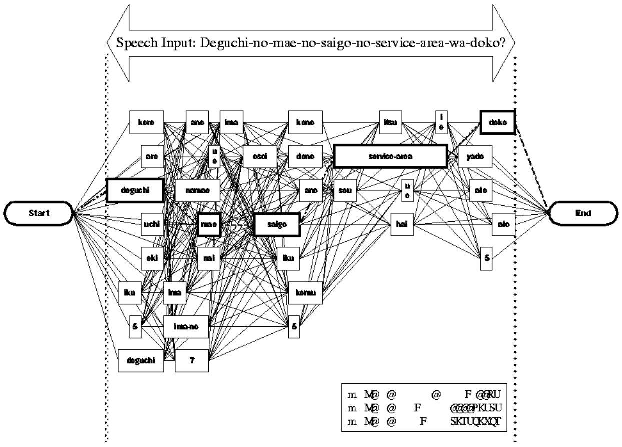

Figure 1: An example of a keyword lattice obtained from the word-spotting engine.

As the vocabulary for speech recognition becomes larger, the number of false alarms in a recognition result increases and the corresponding keyword lattice gets huge. Figure 1 depicts the example keyword lattice which is obtained as the speech recognition result when a user uttered the sentence above ("Deguchi-no-mae-no-saigo-no-service-area-ha-doko?"). 36 keywords, i.e., 5 correct and 31 false alarms, were spotted in this example. When only connectable time constraints are applied, over 4 million plausible word sequences can be generated, each of which is a different path from the start node to the end node, if the lattice is unfolded. Thus a lattice parser, which is capable of parsing without unfolding a given keyword-lattice and generating a set of acceptable word sequence candidates by given grammar, is required.

2.2. Parsing a Keyword-Lattice with Template Hash Tables

Based on the following viewpoints, we have developed the BTH (Bun [meaning "sentence" in Japanese] Template Hash) parser, which is an efficient lattice parser for word-spotting that satisfies the above-mentioned requirements:

Real-time processing in the scale of practical applications on conventional calculation power: Both more vocabulary and less calculation are required to develop a practical speech interface. We aimed to realize a parser working in real-time on 70-80 MIPS RISC MPUs, i.e., the MPUs that are expected to be applied in next-generation car navigation systems.

Grammatical representation taking into account both the recognition method and the application: It is difficult to construct a comprehensive and accurate grammar for actual utterance even if the task to which it is applied is limited, because word order and syntax in actual utterance are wide-ranging. A weakly constrained grammatical representation is required to cope with noticeable phenomena in speech, such as inversions. On the other hand, it is pointless for the parser to accept sentences which the application program cannot execute. A grammatical representation that generates very complicated sentences, e.g., one employing recursive rules, would be too expressive for practical use.

Parsing without unfolding the given lattice: In the case of a large vocabulary, a recognition result lattice from a word-spotting engine includes many false alarms as mentioned above. It costs huge calculation time if the parser analyzes unfolded candidates one by one. Therefore, a lattice parser, which is capable of parsing without unfolding a given keyword lattice, is required.

The BTH parser embodies the following features:

Grammatical representation based on templates of word-class sequences: The grammar is represented as a set of templates, each of which forms a word-class sequence. An interface designer collects sentences in a task and classifies them into sentence-types, i.e., word-class sequences. The parser analyzes a keyword lattice utilizing the template sets. The grammar must be represented non-recursively.

Generation of a hash table from the sentence-type template: As a pre-process of the parsing, the given sentence-type template is compiled into a hash table, whose element is a correspondence between the appearance of a certain word-class in sentences and the established set of sentence-types which applies to the word-class. A Bun (meaning "sentence" in Japanese) Template Hash table is composed of representations, each of which means "if a word of a certain word-class is recognized at a certain order in a sentence, the word of the order applies to either of certain sentence-types."

Parsing of a lattice by propagating belonging sentence-type information: The parser scans the given lattice from the start node. At each node, the set of sentence-types, to which a certain node in the lattice belongs, is calculated by examining set calculations between the corresponding elements in the hash table and the set of sentence-types propagated from the nodes connected just in front of the node. Then the set is propagated from the node to connecting node(s). By repeating the method, the set of sentence-types to which the given lattice belongs is finally obtained.

The detailed parsing process is described in the next subsection.

2.3. Parsing Process

To parse a keyword lattice, the BTH parser utilizes 4 kinds of information structure, i.e., a dictionary of word-classes and words, a dictionary of patterns of word-class sequences, a sentence-type hash dictionary, and a list of processing nodes. The list of processing nodes is the set of nodes that are the current targets of processing. Each node in the lattice holds 3 kinds of data, i.e., a list of antecedent nodes, a list of unprocessed antecedent nodes, and an inter-processing list, composed of sets of sentence-types, each of which is the sentence-type to which the node belongs at a certain order.

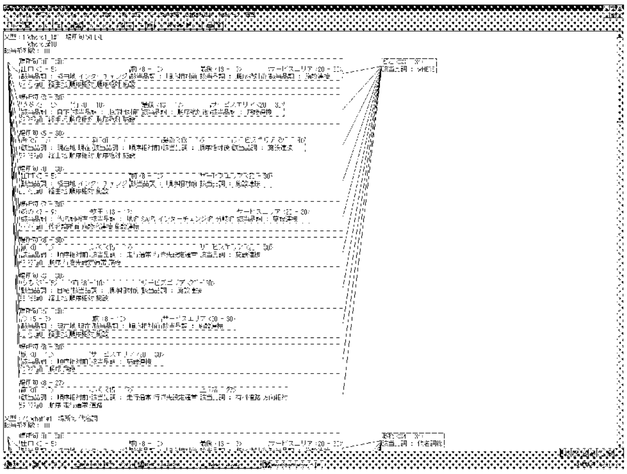

Figure 2: Sample screen of the parser result.

Given a new lattice to analyze, at the beginning of the parsing process, the BTH parser (1) copies the list of antecedent nodes to the list of unprocessed antecedent nodes for each nodes in the lattice, (2) adds all the nodes which may possibly be heads of result word sequences to the list of processing nodes, and (3) sets the initial score that is obtained from the sentence-type hash dictionary for each inter-processing list of the node in the list of processing nodes.

Then the parser repeats the propagation process while the list of processing nodes is not empty. In the propagation process, the following steps are applied to each node in the list of processing nodes:

1. Take one node as the current target from the list of processing nodes and check whether the list of unprocessed antecedent nodes is empty. If it is not empty, check the next node in the list of processing nodes.
2. If the list of unprocessed antecedent nodes of a node X in the list of processing nodes is empty, propagate the inter-processing list of X to all succeeding nodes and add them to the list of processing nodes, and then remove X from the list.
3. At each node to which the inter-processing list of its antecedent node is propagated, re-calculate the inter-processing list.

If the list of processing nodes is empty when the step 3 above is completed, the inter-processing list of the end node is the set of acceptable sentence-types by the lattice. The parser result, i.e., the set of acceptable word-sequences by the lattice, is obtained from the set of sentence-types by re-scanning the lattice. Figure 2 shows an example parser result.

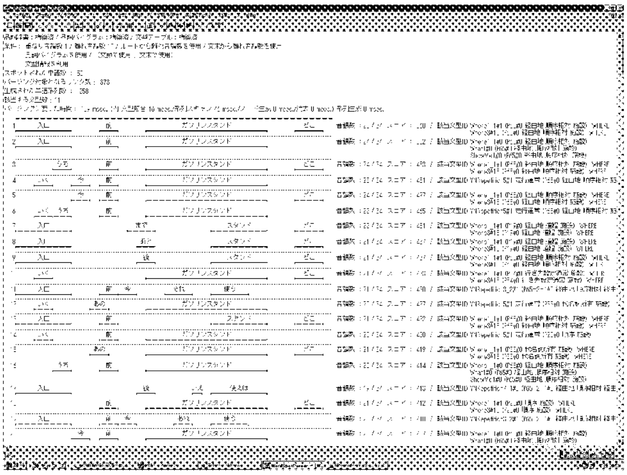

Figure 3: Sample screen of the list of word-sequences.

The cost of the entire parsing process can be estimated from the cost of set calculations to obtain the inter-processing list for each node. The maximum possible number of set calculations, i.e., unions and intersections, is equal to the number of links in the lattice times the number of words in the longest word sequence. It is the worst case, however. Much less calculation is required in the most cases.

3. DIALOGUE SYSTEM

We have implemented a spoken dialogue system using the BTH algorithm, which is able to respond to queries respecting locations made by a user while driving a car. The system is composed of four independent software modules, i.e., the dialogue module that employs the BTH parser, the application module, the word-spotting engine, and text-to-speech synthesizer. The first two modules run on a Windows NT based PC and the other two run on an EWS. The modules run collaboratively, communicating with each other via socket-based messaging. As shown in Figure 4, the application module has a map-based appearance, which is based on ProAtlas, a map software developed by ALPS Mapping Co., Ltd. It simulates car driving by re-playing pre-recorded position data obtained from the GPS (Global Positioning Satellite) system.

The application module simulates driving a car and continuously notifies the dialogue module of the simulated current position at regular intervals of time. When a user asks a question in Japanese, e.g., “Deguchi-no-mae-no-saigo-no-service-area-ha-doko? (Where is the last service area before the exit?)", the word-spotting engine recognizes the utterance and notifies the dialogue module of a word-lattice as depicted in Figure 1.

The dialogue module analyzes the word-lattice by employing the BTH parser and obtains a list of candidate word-sequences, which is sorted in the descending order of the initial score of each candidate, which is calculated from phonetic scores of words of the candidate. Each word-sequence candidate is converted into a user’s intention in the form of a typed feature structure, and is resolved by using current position information and the knowledge base. The knowledge base is a semantic network that contains all the knowledge required to solve questions about locations on the displayed map.

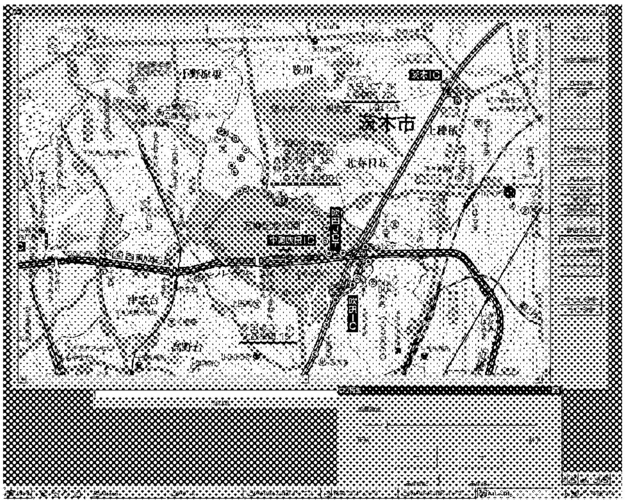

Figure 4: Sample screen of the application.

The score of each candidate is revised taking account of the cost of problem solving, and the list of the candidates is reordered. As a result, the candidate with the best score is selected and the text of the answer corresponding to the question is generated, e.g., “Amagasaki Service Area is the last service area before the Nishinomiya Interchange,” and the dialogue module notifies the synthesizer module of it.

4. CONCLUDING REMARKS

We have designed an efficient keyword lattice parser, BTH, which accepts a user’s spontaneous speech input, and developed a PC-based speech interface. Given a grammar consisting of a set of production rules with neither recursive calls nor loops, the BTH parser performs parsing without unfolding the given keyword lattice. We have tested the parser by applying it to a task of car navigation involving over 700 words of recognized vocabulary. The developed speech interface is able to respond to a user’s question within a few seconds of the question being asked.

In the case of the example task, it is common for over 100 spotted words to be notified from the recognition engine, and consequently over 1 million possible word sequences can be generated by unfolding the corresponding lattice even if word-class bi-gram is applied to the lattice. Our method, however, is able to parse such a large lattice in a practical time. The result shows promise in implementing the function of spontaneous speech interface in practical applications.

REFERENCES

1. Takebayashi, Y., Tsuboi, H., Kanazawa, H., Sadamoto, Y., Hashimoto, H., and Shinchi, H. “A Real-Time Speech Dialogue System Using Spontaneous Speech Understanding”, ***IEICE Trans. Inf. & Syst.,*** E76-D: 112–120, 1993.

2. Takebayashi, Y., Tsuboi, H., and Kanazawa, H. “Keyword-Spotting in Noisy Continuous Speech Using Word Pattern Vector Subabstraction and Noise Immunity Learning”, ***Proc. ICASSP ‘92,*** II–85–88, 1992.

3. Tomita, M. “An Efficient Word Lattice Parsing Algorithm for Continuous Speech Recognition”, ***Proc. ICASSP ‘86,*** 1569–1572, 1986.