Journal of Computer Science and Technology, pp. 924-932, Vol.17, No.6, November 2002

SPONTANEOUS SPEECH PARSING IN TRAVEL INFORMATION INQUIRING AND BOOKING SYSTEMS

Pengju Yan, Fang Zheng, Hui Sun, and Mingxing Xu

Center of Speech Technology, State Key Laboratory of Intelligent Technology and Systems
Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, China
[*yan, fzheng, sunh, xumx*]*@sp.cs.tsinghua.edu.cn*, http://sp.cs.tsinghua.edu.cn

ABSTRACT

Grammar-based parsing is a prevalent method for natural language understanding (NLU) and has been introduced into dialogue systems for spoken language processing (SLP). A robust parsing scheme is proposed in this paper to overcome the notorious phenomena, such as garbage, ellipsis, word disordering, fragment, and ill-form, which frequently occur in spoken utterances. Keyword categories are used as terminal symbols, and the definition of grammar is extended by introducing three new rule types, *by-passing*, *up-messing* and *over-crossing*, in addition to the general rules called *up-tying* in this paper, and the use of semantic items simplifies the semantics extraction. The corresponding parser *marionette*, which is essentially a partial chart parser, is enhanced to parse the semantic grammar. The robust parsing scheme integrating the above methods has been adopted in an air traveling information service system called *EasyFlight*, and has achieved a high performance when used for parsing spontaneous speech.

KEYWORDS: Spoken Language Understanding, Spoken Dialogue System, and Spontaneous Speech

1. INTRODUCTION

Spoken language understanding (SLU) is one of the most significant parts in spoken language understanding and dialogue systems. The performance of the SLU component greatly affects the performance of spoken dialogue systems. Currently, grammar-based parsing is the most popular approach used in the area of SLU and there exist two variations. One is based on the continuous speech recognition where the run-through utterances are fully recognized for future use [1]. Another uses the keyword/concept based technology where only meaningful and promising speech parts are considered [2,3]. It is known that in dialogue systems the users' utterances are very casual and are full of un-grammatical phenomena, such as garbage, fragment, hesitation, correction, repetition, ellipsis, word disordering and ill form. Apparently the first approach will be fatally broken down because the required complete syntactic results will be hardly achieved in the presence of these phenomena [4].

Furthermore, while English is a well-structured language [5], Chinese, along with its colloquial form, is an ideographic language to a great extent. That is to say, English-inspired approaches such as that proposed in [1], based on the assumption that the words in any sentence follow a rigid order, will not work satisfyingly in Chinese spoken dialogue systems.

Therefore, SLU experts are devoting more time to searching for more robust strategies for the second approach. Kono, Yano and Sasajima have presented a parsing algorithm for word spotting [6], which can efficiently parse a keyword lattice containing a large number of false alarms. They have also developed a generic framework for developing spoken dialogue systems [2] where they use the keyword-spotting method to extract plausible word sequences by ignoring the misuse or loss of particles and unnecessary terms such as "*aah*" or "*well*".

Partial parsing is a widely used technique to deal with the spoken language phenomena and speech recognition errors. Based on it, Boros and Heisterkamp define a phrase-spotting method and use an agenda-driven island-based active chart parser [7] where the reduction can be performed across gap words. Furthermore, Noth and Boros state that [3] they restrict the linguistic analysis to the semantic concepts, which results in several grammar fragments rather than one full grammar, and the island-based parsing technique has been proved quite robust against spontaneous speech phenomena.

There are a lot of hybrid instances of the methods mentioned above. Three major advantages of these methods are as follows. (1) They deal with only the semantically meaningful parts of the input utterance and thus the garbage and domain meaningless parts are bypassed. (2) The parser can combine constituents by skipping irrelevant parts in between. (3) Every partial parsing result is maintained instead that one complete tree result and one null result are the only two choices that can be achieved. However, they lack a systematic way to cope with other speech phenomena such as repetition, word disordering, and are not prone to be used by the semantics-extraction component.

To provide the solution, a robust parsing scheme is proposed in this paper, where there are four different types of grammar rules, and the parser, an enhanced chart parser embedded with multiple control strategies, applies the grammar to the utterances. The parsing scheme is adopted in our air travel information service system *EasyFlight* and the satisfying parsing results are achieved.

2. ROBUST PARSING SCHEME

Generally speaking, in a large number of the speech understanding systems, N-grams are used as the language model in recognizers to produce N-best word strings, and a word-class model is employed to determine the inner part-of-speech (POS) strings. POSs are used as the terminal symbols for the transcription of grammar, thus the nodes of the parse tree are pure within the syntactic category. However, we argue that these methods are not efficient for spoken language processing because of the following reasons. (1) A smoothed N-gram model is usually not easy to get in the absence of a sufficient corpus. (2) A general POS-based syntactic grammar is unnecessary for the narrow domain in a dialogue system and is also hard to write. (3) This kind of grammar is too rigorous for use in the presence of a large number of ungrammatical sentences, i.e., not robust enough. (4) It is not efficient/convenient to achieve semantic representations from only syntactic trees.

On the other hand, according to our analysis on the domain-specific corpus, semantic items rather than syntactic ones are easy to be modeled in the cases of various kinds of ungrammaticalities.

To provide our solutions in this paper, we propose a robust parsing scheme where the keywords/fillers are the basic units for speech recognition, the grammar is extensively defined to accommodate four types of rules. Keyword categories and semantic items are used as the grammar symbols. An enhanced chart parser is used to parse the spontaneous speech. A semantic function tree translates the parse tree into semantic frames. All these above methods are expected to be efficient for dialogue systems, especially in the situation that the spontaneous linguistic phenomena cover almost all parts of the utterances.

2.1 Corpus analysis

We collected domain-specific corpus through a multi-channel telephone recording system, which was placed in an air travel agency to monitor the real-world conversations. The main objective of the recording project is to collect various conversation phenomena and styles, thus we monitored only the telephone lines regarding domestic airline dialogues. The final corpus contains more than 6 gigabytes' (200 hours) speech data in the PCM format, and about 3 gigabytes' (or 100 hours') valuable parts of which, with undistinguished parts discarded, have been transcribed at the Chinese character (text) level. Analysis on the transcription shows there occur large amounts of ungrammatical sentences, and most of which can attribute to the notorious spontaneous linguistic phenomena. The detailed classifications of the phenomena as well as the corresponding examples are given as the follows.

- The courtesy items / sentences inessential for semantic analysis.
  C: 喂，你好，请问是中关村航空客运代理处么?
  hi hello could you tell me
- Repetitions because of pondering or emphasizing when speaking.
  C: 我问一下那个四月三十呃四月三十号北京到...
  30th April 30th April
- Ellipsis in the context.
  C: 我问一下那个四月三十呃四月三十号北京到福州的机票最后一班还有么? (Departure city, arrival city, and departure time provided while requesting the appropriate flight)
  O: 只有一一班有。("Only one flight available")
  C: 那个那五月一号的下午三点有么? (Only departure time updated leaving other information unchanged)
- Constituents appearing in any order (as long as sufficient information is given).
  C: ...五点二十五国航飞深圳的... (Time, airline code, location and some other items can appear in any order)
- Parols (verbal idioms) or unnecessary terms.
  C: 那，那个八点二十那个是去什么机场的呀？ ("那"/"那个" is somewhat similar to "uhm")
- And long sentences with all required information.
  C: 哎，您好，这样那个我订一张(one)那个明天(tomorrow)下午(afternoon)五点四十五(5:45)去北京(from Beijing)到上海(to Shanghai)的那个机票(ticket)的。

All these spontaneous speech phenomena are the greatest challenges to Mandarin dialogue systems.

2.2 Keyword List

If only the semantically relevant speech segments are to be taken into account, we can use a comparatively small grammar to achieve a comparatively large coverage. This is the major motivation for using keyword categories as terminal symbols. In addition, using keywords can also simplify the speech recognition when no N-gram model is available.

By browsing the corpus, we design a keyword lexicon with approximately 600 words which is divided into about 70 categories. These categories can be clustered into three classes according to their functions as depicted in Table 1.The prefix *mat_* denotes the *material* class, the prefix *tag_* the *tag* class, and the prefix *ato_* the *atom* class. The *material* class contains the categories that describe the key information about a flight, including time, city name, airline code (airways), etc. Keywords of class *tag* are usually the interrogatives, prepositions, pronouns and some special semantic markers in *EasyFlight*. Keywords in class *atom* are some morphemes, such as numbers and word prefixes/suffixes.

Table 1. Examples of the keyword categories

Categories	Examples and Explanation
mat_city_name	"北京" ("Beijing")
mat_airline_code	"CA" ("Air China")
mat_aircraft_type	"波音 747" ("Boeing 747")
mat_time_of_the_day	"上午" ("morning")
tag_from_here	"从这儿" ("from here")
tag_to	"到" ("to")
tag_exist_or_not	"有没有" ("exist or not")
tag_how_many	"多少" ("how many")
ato_week	"礼拜" (weekly-date prefix)
ato_january_prefix	"元" (January prefix)
ato_0to9_yao	"一" (digits for ID spelling)
ato_1to6	"六" (digital suffix for weekly-date)

In the current version, more than half of the total keywords, say 340 of 600, are of the *material* class, including 150 city names, 60 airline codes (airline company abbreviations), 60 airline company names, 50 aircraft types, and 20 date words. Another 100 keywords are of the *tag* class, and the rest are of the *atom* class. The keyword categories are practically used as the terminal symbols to write the grammar, while the keyword class names presented here are only abstract concepts and not for actual use.

2.3 Definition of Grammar

As mentioned above, utterances in dialogue systems are full of spontaneous speech phenomena. In this case, traditional grammars where word-classes or part-of-speeches are taken as terminal symbols, with which linguists are quite familiar, will not work well because a great deal of daily spoken sentences will be rejected due to the narrow coverage of the grammars. At this point, we make use of a grammar in which the terminal/non-terminal symbols are all semantically meaningful constituents; therefore we call it a *semantics-based grammar* or *semantic grammar* in brief.

Additionally, we enhance the context-free-grammar (CFG) to extend the coverage by introducing a property named *type* for each rule of the grammar to indicate the arc extension strategy applied in run-time parsing. The definitions of the four types, *up-tying* (*→), *by-passing* (→), *up-messing* (@→), and *over-crossing* (#→), are given here in detail.

- **Definition Ⅰ.** [sentence] A sentence (utterance) is a string of keyword categories or fillers, $sent = (K_0, K_1, \cdots, K_{n-1})$, where n denotes the length of the sentence, and K_i the i_{th} keyword category or filler, $0 \le i < n$. □
- **Definition Ⅱ.** [position] The position of a terminal constituent K_i in sentence $(K_0, K_1, \cdots, K_{n-1})$ is defined as $[i, i+1)$, where $0 \le i < n$. And the position of a non-terminal constituent C is defined as $P_C = [p_1, p_2)$, where $p_1 = \min\{b \mid [b, e) \text{is the position of a leaf node of } C\}$, and $p_2 = \max\{e \mid [b, e) \text{is the position of a leaf node of } C\}$. □
- **Definition Ⅲ.** [occupation, conflict or overlap with each other] The occupation of a terminal constituent K_i is $o_i = \{i\}$. The occupation of a non-terminal constituent C is $O_C = \{o \mid o \text{ is the occuptation of a leaf node of } C\}$. Two occupations O_1 and O_2 are called conflicting with each other iff. $(\exists l \in O_1, \exists m \in O_2, l = m)$, and are called overlapping with each other iff. $(\exists i, 1 \le i \le 2, (\exists l, m \in O_i, \exists n \in O_{2-i}, (l < n < m)))$. □

If a grammar rule is written as $Y\,[rule\_type] \rightarrow Y_0 Y_1 \cdots Y_{m-1}$, C_j is the instantial constituent of Y_j, $P_j = [p_{j,1}, p_{j,2})$ is the position of C_j, and O_j is the occupation of C_j, where $0 \le j < m$, then the four rule types can be defined as follows.

- **Definition Ⅳ.** [up-tying type] An up-tying rule where $rule_type = $'*' stands for that for $\forall \le <$, C is not an over-crossing constituent, and $(\forall 0 \le j < m-1, p \quad = p_{+})$. □
- **Definition Ⅴ.** [by-passing type] A by-passing rule where $rule\_type = \phi$ stands for that for $\forall \le <$, C is not an over-crossing constituent, and $(\forall 0 \le j < m-1, p \quad \le p_{+})$. □
- **Definition Ⅵ.** [up-messing type] An up-messing rule where $rule_type = $'@' stands for that for $\forall \le <$, C is not an over-crossing constituent, and $\forall 0 \le j,k < m,\ j \ne k$, O_j and O_k do not overlap with each other. □

- **Definition Ⅶ**. [over-crossing type] An over-crossing rule where $rule\_type = '\#'$ stands for that for $\forall 0 \le j,k < m,\ j \ne k$, O_j and O_k do not conflict with each other. □

An *up-tying* type rule is a conventional rule used in conventional CFG grammars. By using a *by-passing* type rule, constituents can be reduced by skipping irrelevant segments. An *up-messing* or an *over-crossing* type rule will be helpful to group constituents despite the order they occur. One difference between the last two types is that an *up-messing* rule does not contain any *over-crossing* sub-constituents. Another difference is that, the latter type of rules are used to reduce sub-constituents no matter whether their occupations overlap with each other or not, while the former one not.

2.4 Transcription of Semantic Grammar

Though in some literature semi-automatic generation methods were reported (e.g. [8]), we generate the grammar manually because the transcription effort is greatly alleviated using our approach. We present here some rule examples in the domain of *EasyFlight* to demonstrate how the four rule types are chosen in case of various situations.

2.4.1 Up-tying rules

The *up-tying* rules are needed in at least one case when the customer's ID card no. is to be parsed where the ID card no. is taken as a crucial piece of information forbidden to be inserted by or mixed with other terms.

sub_id_card_head *→ *ato_0to9_yao ato_0to9_yao ... ato_0to9_yao* (15 identical terms)
id_card_no → *sub_id_card_head*
id_card_no *→ *sub_id_card_head ato_0to9_yao ato_0to9_yao ato_0to9_yao*

2.4.2 By-passing rules

A large number of rules are of the *by-passing* type, which is based on the assumption that the input keyword string is full of recognized fillers/rejections, speech fragments or some other nonsense parts. E.g., "星期啊三嗯星期四" ("week ah three en week four"/Wedn-ah-esday and ah Thursday) is admitted if the following *by-passing* rules exist.

sub_week_day → *ato_week ato_1to6*
sub_week_day_list → *sub_week_day*
sub_week_day_list → *sub_week_day sub_week_day_list*
sub_date → *sub_week_day_list*

2.4.3 Up-messing rules

The *up-messing* rules are required in case no matter what order sub-constituents may follow. In *EasyFlight*, constituents of *time*, *location*, and *plane type* can be grouped by *up-messing* rules since the user can tell them in any order.

timeloc_info_cond @→ *info_date_time_cond info_fromto*
plane_info @→ *mat_airline_code mat_aircraft_type*
flight_info_cond @→ *timeloc_info_cond plane_info*

2.4.4 Over-crossing rules

Some concepts, which can be defined as the task-relative minimal elements, may be derived from several different *by-passing* rules and can be used to form other constituents. *Over-crossing* rules are used to avoid the definition of many similar rules in this case. E.g., "是不是 (be or not) *confirm_c*" , "*confirm_c* 是不是", "*confirm_c* 是吗 (be or not?)", and "是(be) *confirm_c* 吗 (*question mark*)" can be described by a single *over-crossing* rule.

mark_q_is → *tag_is_or_not*
mark_q_is → *tag_is tag_question_mark*
mark_q_is → *tag_is_q*
confirm_request #→ *mark_q_is confirm_c*

The grammar is transcribed following the aforesaid ideas, and totally there are about 200 rules were written for *EasyFlight*, and most of them are *by-passing* rules. The coverage of the grammar is proven to be wide enough in the system, and the semantic extraction can be performed directly because the concepts, e.g. *mark_q_is* in section 2.4.4, are formalized in the rules.

2.5 *Marionette* Parser: an Enhanced Chart Parser

In our parsing scheme, an enhanced chart parser is used to parse the input sentence. In addition to the inherent characteristics of a chart parser that all partial results are maintained, there are some more control strategies embedded in the parser, which are

- combining the nonadjacent sub-constituents by skipping other constituents for *by-passing* rules;

- considering all the possible occurring order of the focused sub-constituents in the *up-messing* rules;
- grouping the sub-constituents whether their occupations overlap with each other or not in the *over-crossing* rules; and
- ranking the constituents according to several key criteria for disambiguation; etc.

Before the detailed parsing algorithm being discussed, some definitions must be given beforehand.

- **Definition VIII.** [position of active arcs] The position for an active arc $Y\left[rule\_type\right]\rightarrow Y_0Y_1\cdots Y_{l-1}{}^{\circ}\cdots Y_{m-1}$ is defined as $P_Y=\left\lfloor p_1=\min\left\{b_j, 0\le j<l\right\}, p_2=\max\left\{e_j, 0\le j<l\right\}\right\rfloor$, where $\left\lfloor b_j, e_j\right\rfloor$ is the position of Y_j, $0\le j<l$. □
- **Definition IX.** [occupation of active arcs] The occupation for an active arc $Y\left[rule\_type\right]\rightarrow Y_0Y_1\cdots Y_{l-1}{}^{\circ}\cdots Y_{m-1}$ is defined as $O_Y=\bigcup_{0\le j<l} O_{Y_j}$, where O_{Y_j} is the occupation of Y_j, $0\le j<l$. □

A part of the algorithm for the arc extensions of the four types of rules is given in Figure 1. Here the four types of active arcs are extended by accurately following the above-mentioned definitions of the grammar.

For constituent *C* at position (p_1, p_2):

a) for each active arc $Y*\rightarrow Y_1Y_2\cdots Y_k{}^{\circ}C\cdots Y_m$ at position (p_0, p_1'), if $p_1'= p_1$, add a new active arc $Y*\rightarrow Y_1Y_2\cdots Y_kC^{\circ}\cdots Y_m$ at position (p_0, p_2);

b) for each active arc $Y\rightarrow Y_1Y_2\cdots Y_k{}^{\circ}C\cdots Y_m$ at position (p_0, p_1'), if $p_1'\le p_1$, add a new active arc $Y\rightarrow Y_1Y_2\cdots Y_kC^{\circ}\cdots Y_m$ at position (p_0, p_2);

c) for each active arc $Y@\rightarrow Y_1Y_2\cdots Y_k{}^{\circ}Y_{k+1}\cdots Y_{l-1}CY_{l+1}\cdots Y_m$, if the occupations of *C* and the arc do not overlap with each other, add a new active arc $Y@\rightarrow Y_1Y_2\cdots Y_kC^{\circ}Y_{k+1}\cdots Y_{l-1}Y_{l+1}\cdots Y_m$ at the calculated actual position;

d) for each active arc $Y\#\rightarrow Y_1Y_2\cdots Y_k{}^{\circ}Y_{k+1}\cdots Y_{l-1}CY_{l+1}\cdots Y_m$, if the occupation of *C* and the arc do not conflict with each other, add a new active arc $Y\#\rightarrow Y_1Y_2\cdots Y_kC^{\circ}Y_{k+1}\cdots Y_{l-1}Y_{l+1}\cdots Y_m$ at the calculated actual position.

Figure 1. A part of the parsing algorithm for the arc extensions

When ambiguities are encountered, the most promising/reasonable tree must be selected. Some measures and decisions are designed to rank the competing constituents, which can be listed in Table 2 according to the applied priorities.

Table 2. Ranking constituents

Higher	Lower
Top-level constituents (non-sub-constituents)	Non-top-level ones
Non-top-level constituents ranking the same	
Constituents with larger coverage (the number of the occupation set) and larger acoustic score	Ones with smaller coverage and smaller acoustic scores
Constituents reduced from top rules (the left symbol of which does not appear as a right symbol of any other rules)	Ones reduced from non-top rules
For constituents reduced from top rules, ones with smaller number of nodes (the nodes in the tree of the constituent)	Ones with larger number of nodes
For constituents reduced from top rules, ones with smaller depth values (the depth of the tree of the constituent)	Ones with larger depth values
Constituents reduced from non-top rules ranking the same;	
Constituents with larger occurrence value (the sum of all the elements in the occupation set)	Ones with smaller occurrence value
Constituents reduced later	Ones reduced earlier

The constituent with the highest rank, along with all of its descendant nodes, is taken as the most promising/reasonable one and will survive. Other top-level constituents will also survive, as long as they do not conflict with the pre-determined reserved ones. Finally a parsing tree or several parsing trees (thus compromise a parsing forest) will be maintained as the ultimate result.

3. APPLICATION AND EVALUATION

The parsing scheme is applied to *EasyFlight* and some primary evaluation is made. There are four functional modules in *EasyFlight*, as depicted in Figure 2. The keyword spotter produces the N-best keyword strings. The *marionette* parser processes the keyword strings and output the resultant tree/forest.

We propose a semantic-function-trees mechanism to analyze the semantics. Basically an interpreter function, which determines how to fill the semantic frame according to the constituent concerned, is statically appended to each grammar rule, hence each constituent node of the resultant parsing tree/forest is dynamically associated with an interpreter function. In this manner, each parse tree is accompanied with a tree of semantic interpreter functions, and the two trees are isomorphic to each other. The semantics interpretation is performed by means of calling the root interpreter functions and the middle/terminal functions are

called iteratively if necessary.

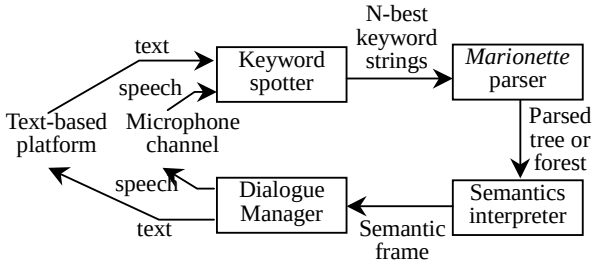

Figure 2. *EasyFlight* modules

Currently the parsing scheme is applied to a text-based instance as well as a microphone channel instance of *EasyFlight*. Primary experiments on *EasyFlight* show that the parsing scheme achieves a satisfying robust parsing performance. The speech phenomena, such as acoustic garbage, linguistic garbage, repetition, ellipsis, word disordering and ill form, are overcome efficiently. Here several parsing examples involved in the aforementioned phenomena are illustrated as follows.

- A sentence with the acoustic or linguistic garbage/filler is parsed as shown in figure 3. The garbage/filler can occur in any place in the sentence, and by using of *by-passing* rules they are ignored in the constituent-composing process.

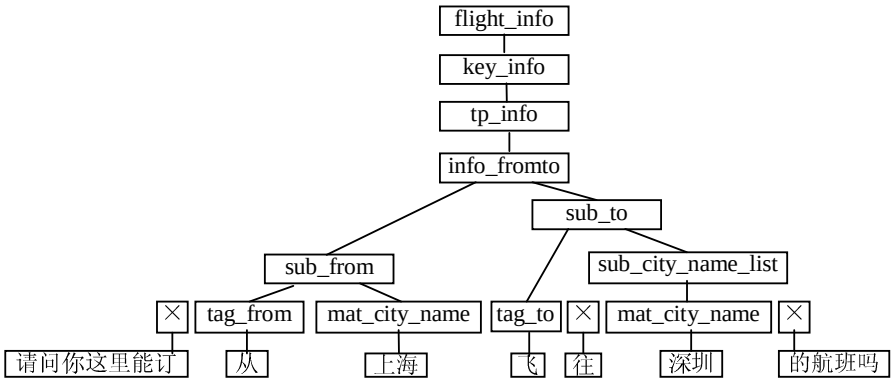

Figure 3. Example with garbage

- In figure 4 a parsing instance example with repetition is depicted. In this example, the same departure city (北京/Beijing) occurs twice. According to our pruning/optimizing strategies described in Section 2.5, the resultant tree that contains the latter “Beijing” is preserved.

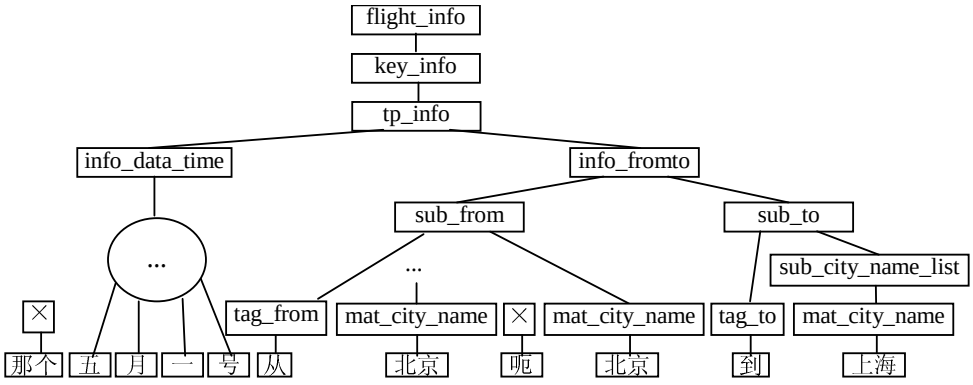

Figure 4. Example with repetition

- The word/phrase order variations are also covered in our methods. Also see Figure 4, if the user alter the information order of time and city names, a similar tree can be obtained too, except that the sub-nodes of *tp_info* will change over.
- Since only crucial speech/language parts are concerned in our scheme, all important information will be retained no matter how little or how much information the user provides, and no matter whether the sentence is *syntactically legal* or not. The

corresponding trivial examples are not given here for the sake of compactness.

Such kinds of phenomena in *EasyFlight* are very common, and the solution to these problems greatly enhances the performance of the whole system. A formal evaluation was performed on a small corpus collected from the students of our laboratory, which contains 100 sentences-in-domain, including 58 questions and 42 statements. The corpus was parsed in 17.13 seconds on a computer equipped with an Intel Celeron 466 CPU. Basically, 78 sentences were successfully parsed, where the term "successfully" means that the main part of sentence was successfully parsed so that the dialogue manager can get enough derived information to continue the discourse. The rest 22 sentences failed to pass through the parser. Thus the accuracy of the parser is 78%.

Table 3. Reasons of parsing failure

Description	**Reason**	**Count**
Parsed as multiple sentences or fragments	Not covered by the grammar	10
Concepts missed or mistaken	Not covered by the grammar	12
Concepts mistaken	Parsing strategy	1
Others	Not covered by the lexicon, or wrong sentence	4

The overall errors are classified in Table 3, where the errors included those occurred in successfully parsed sentences. It is clear that if the lexicon and the grammar are properly revised, most of the errors can be overcome.

4. CONCLUSIONS AND DISCUSSIONS

In this paper a parsing scheme is presented, the goal of which is to parse the spontaneous speech robustly. The definition of the grammar is extended to accommodate four types of rules and the constituents are within the concept/semantic category other than the syntactic category. An enhanced chart parser *marionette* employs the semantic grammar to parse the sentences, and eliminates ambiguities by pruning and optimizing.

The parsing scheme is adopted in a dialogue system *EasyFlight* where there are about 600 keywords and 70 keyword categories and the grammar size is 200.

The main advantage of the proposed parsing scheme is that the semantic items instead of the syntactic items are used at the stages from recognition to parsing, thus the domain irrelevant speech/language parts are ignored to "highlight" the real concerned and crucial speech/language parts. The use of semantic grammar amplifies that effect, and the semantic interpretation is performed directly in this manner.

Some may argue that a general syntactic grammar can achieve a higher coverage when switching to another domain. But as we all know that it is so hard to write a perfect general grammar that few successful instances have come out till even today, and naturally a more general grammar requires a more complicated parser. But, our methods are easier to use and less time-consuming in domain-specific applications, both in terms of system designing efforts and run-time complexities. Furthermore, both syntactic and semantic analyzing are unified in this manner.

However, there are still some problems with our methods: How to efficiently use the grammar to guide keyword spotting? How to efficiently derive the keyword list and grammar from the corpus when the corpus or the designers' strategy changes? And how to efficiently parse the speech when the grammar is complicate?

We plan to design a customized platform to provide a convenient and systematic way for system designers to customize the keyword list, the grammar, and the keywords/rules subset at different dialogue states. Priorities will be associated with each rule and parsing will be done in all stages. All these methods are expected to fix the problems mentioned above.

5. REFERENCES

[1] S. Seneff, "TINA: a Natural Language System for Spoken Language Applications", *Computational Linguistics*, Vol. 18, No. 1, 61-86, 1992.

[2] M. Sasajima, T. Yano, and Y. Kono, "EUROPA: A Generic Framework for Developing Spoken Dialogue Systems", *EuroSpeech'99 Proceedings*, Budapest, Page 1163-1166, 1999.

[3] E. Noth, M. Boros, J. Haas, V. Warnke, and F. Gallwitz, "A Hybrid Approach to Spoken Dialogue Understanding: Prosody, Statistics and Partial Parsing", *EuroSpeech'99 Proceedings*, Budapest, Page 2019-2022, 1999.

[4] V. Zue, "Conversational Interfaces: Advances and Challenges", *EusoSpeech'97 Proceedings*, Rhodes - Greece, Keynote Speech, 1997.

[5] I. Schadle, J.-Y. Antoine, and D. Memmi, "Connectionist Language Models for Speech Understanding: The Problem of Word Order Variation", *EuroSpeech'99 Proceedings*, Budapest, Page 2035-2038, 1999.

[6] Y. Kono, T. Yano, and M. Sasajima, "BTH: An Efficient Parsing Algorithm for Word-Spotting", *ICSLP'98 Proceedings*,

Sydney, 1998.
[7] M. Boros, and P. Heisterkamp, “Linguistic Phrase Spotting in a Simple Application Spoken Dialogue System”, *EuroSpeech’99 Proceedings*, Budapest, Page 1983-1986, 1999.
[8] K. C. Siu, and H. M. Meng, “Semi-Automatic Acquisition of Domain-Specific Semantic Structures”, *EuroSpeech’99 Proceedings*, Budapest, Page 2039-2042, 1999.