Complex 2004

Proceedings of the 7th Asia-Pacific
Conference on Complex Systems
Cairns Converntion Centre, Cairns, Australia
6-10th December 2004

Interactive Genetic Programming for the Sentence Generation of Dialogue-based Travel Planning System

Sungsoo Lim, Kyoung-Min Kim, Jin-Hyuk Hong and Sung-Bae Cho
Dept. of Computer Science, Yonsei University
134 Sinchon-dong, Sudaemoon-ku
Seoul 120-749, Korea
Email: {lss, kminkim, hjinh}@sclab.yonsei.ac.kr, sbcho@cs.yonsei.ac.kr

Abstract

As dialogue systems have been widely investigated, the research on natural language generation in dialogue has aroused interest. Contrary to conventional dialogue systems that reply to the user with a set of predefined answers, a newly developed dialogue system generates them dynamically and trains answers to support more flexible and customized dialogues with humans. The paper proposes an evolutionary method for generating sentences using genetic programming. Sentence plan trees, which stand for the sentence structure, are adopted as the representation of genetic programming. With interactive evolution process with the user, a set of customized sentence structures is obtained. The proposed method applies to a dialogue-based travel planning system and the usability test demonstrates the usefulness of the proposed method.

1. Introduction

Natural language generation, one of important topics in dialogue systems, automatically creates natural sentences based on a non-linguistic information representation (Theune M., 2003). Current research on dialogue systems has mainly focused on understanding the input query or controling the dialogue flow rather than generating dynamic sentences (Zue V. & Class J., 2000). Canned scripts or templates are usually employed to support answering in those systems, hence they tend to be static, using the same or similar response patterns in the interation with users.

In order to support a flexible and realistic interaction with humans, resent dialogue systems attempt to adopt a mechanism generating sentences. The traditional natural language generation is usually based on grammars or templates. The developer needs to design a proper grammars to prevent creating wrong sentences, or to construct a large set of templates to support various types of sentences. The approach requires the developer to design a number of grammars or templates for every possible sentence structures, and this may be impractical for domains in which many sentence structures are possible (Ratnaparkhi A., 2002).

In order to overcome the limitations, trainable approaches have been attemped in recently (Ratnaparkhi A., 2002; Walker M. A. et al., 2002; Giuseppe R. & Allen L., 2000; Bulyko I. & Ostendorf M., 2002). Walker et al. proposed the sentence plan tree representing the structure of a sentence while training is conducted by a RankBoost method (Walker M. A. et al., 2002). A stochastic model generating sentences was presented by Levin et al. while they have applied it to a dialogue-based travel planning system (Giuseppe R. & Allen L., 2000). Ratnaparkhi has suggested a hybrid model (Ratnaparkhi A., 2002), while Bulyko and Ostendorf generated various sentences using a weighted finite state machine (Bulyko I. & Ostendorf M., 2002). These approaches were attemped in order to construct a dynamic dialogue system that adjusts to the domain.

In this paper, we propose an evolutionary method which makes diverse and flexible scentences in dialogue-based system. It uses an interactive genetic programming that adopts the sentence plan tree as the representation of individual sentences. The method composes a sentence by combining basic units with several joint operators, and iteratively interacts with humans to adjust itself to a domain. A dialogue-based travel planning system is constructed with the proposed method for demonstration, and several senarios and experiments show the usefulness of the proposed method.

2. Natural Language Generation

The study on natural language is largely progressed as natural language understanding (NLU) and natural language generation (NLG). The main goal of NLG is to investigate how computer programs can produce a high-quality natural language text from internal representations of information (Weiwei W. et al., 1996; McKeown K., 1986). In recent years, the increasing feasibility of human-computer dialogue systems have prompted the need for better responses by generating diverse sentences. Dialogue systems that support complex planning are being developed, and these are likely to require more sophisticated system outputs.

The stages of NLG for a given application are disciplined in the order of text planning, sentence generation and speech synthesis. Generation techniques can be classified into four main types: canned text systems, template systems, phrase-based systems, and feature-based systems (Adorni G. & Zock M., 1996). Canned-text systems constitute the simplest approach for generating a sentence or multiple sentences. They are trivial to create, but inflexible and repetitive. Template systems, a little bit more sophisticated than canned-text systems to support flexible interactions, rely on the pre-defined templates or schemas. Therefore, the performance of the system is determined by the quality and quantity of the templates. It usually applies to multi-sentence generation, particularly in applications whose texts are fairly regular in the structure. Many current research have employed template-based generation, because it is conceptually fair and easy to produce high quality outputs that are specific to each dialogue situation. However, it has little reusability and also there are some restrictions when rare explicit representation of linguisitc structures are presented (Seneff S. & Polifroni J., 2000). Phrase-based systems consider generalized templates. In such systems, a phrasal pattern is first selected to match the top level of inputs, and each part of patterns is recursively expanded into a more specific phrasal pattern matching some sub-portion of inputs. At the sentence level, the phrases resemble phrase structure grammars, while at the discourse level they play the role of text plans. Feature-based systems, which are restricted yet to generate a single sentence, represent each possible minimal alternative of expressions as a feature. Accordingly, a sentence is specified by a unique set of features.

Most NLG systems use grammars for generating sentences like parsers with semantic or syntactic information. Hence, rule-based systems or template-based systems are usually utilized. Rule-based techniques need to be richly specified with features. For a rule-based system being more efficient than a template system, the amount of time and effort that go into designing and adhering to such a rich input representation must be justified (Oh H. & Rudnicky I., 2002). One of the greatest advantage of the rule-based technique, however, is that most rules are domain independent, so once developed, they can be reused in many applications by conforming input specifications. NLG might promise the portability between domains and dialogue situations by developing general and domain-independent rules. However, the quality a specific domain or situation may be inferior to that of a template-based system without considerable investment in domain-specific rules or domain-tuning of general rules (Walker M. A. et al., 2002). This is probably why most spoken dialog systems use templates rather than grammar rules.

Adapting a dialogue system to a domain is one of main issues in NLG, and many training methods have been developped for it. Especially, mechine learning techniques have the potential for this problem as attempted in several works (Bulyko I. & Ostendorf M., 2002).

3. The Diverse Sentence Generation using Genetic Programming

Genetic programming is proposed by John R. Koza to create automatically a computer program solving a problem. It is an extension of the genetic algorithm in which each individual in the population is a computer program (Koza J., 1994). In this paper, it generates sentences dynamically and domain-adaptively

An object represented as a sentence plan tree (SPT), denotes a complex sentence. In each SPT, each leaf node contains one simple sentence that is corresponding to the unit and parent nodes represent joint operators for combining child nodes. Figure 1 briefly shows the outline of dynamic sentences generation process using genetic programming.

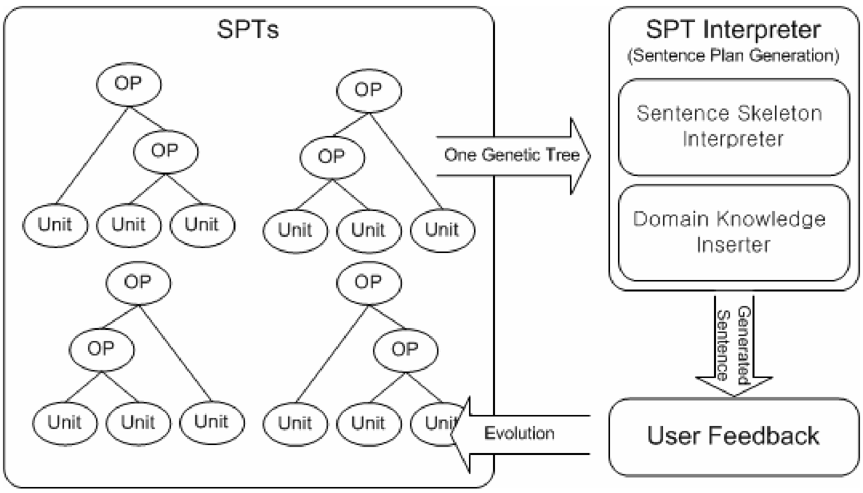

Figure 1. The procedure of generating sentences by interactive genetic programming.

3.1 Representation, Crossover and Mutation

There is one simple sentence corresponding with each units at a leaf node of SPTs in Figure 1, and each internal node means a "joint operator". The simple sentence and the joint operator are defined as follows:

- **simple sentence (SS)**: one sentence which has minimum information, which plays a role of sentence primitives. A simple sentence consists of one subject, one communicative act and one verb.
- **communicative act (CA)**: information which is obtained from target domain (ex. location, time, and so on). When the agent has the information about the communicative act, the simple sentence will become a statement, otherwise it will become a question
- **joint operator (JO)**: a conjunctive operator which connects two sentences. Details are given below.

The SPT interpreter in Figure 1 derives a complex sentence from a SPT. The sentence skeleton interpreter makes a skeleton of a complex sentence by combining simple sentences which appears at the leaf nodes. Then the domain knowledge inserter completes the complex sentence with domain-relevant knowledge, subjects and verbs corresponding with it. Figure 2 presents the procedure of interpreting a STP in the sentence skeleton interpreter. [dLocation] and [dTime] represent slots where information about departure location and time can be filled into.

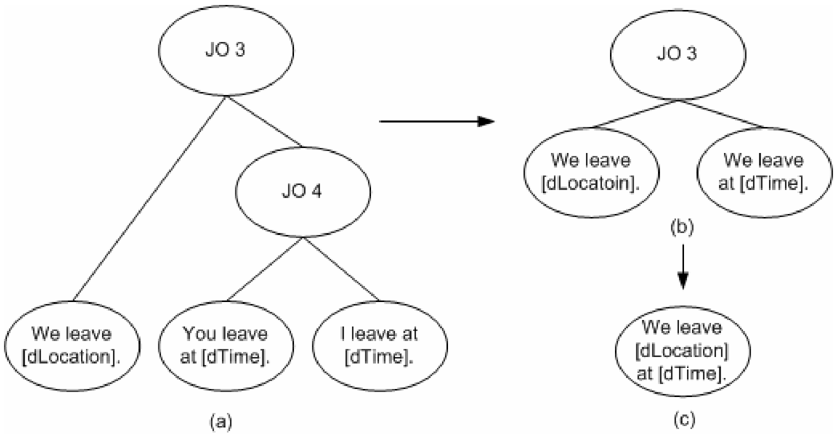

Figure 2. A procedure of interpreting a STP

The STP in Figure 2 (a) becomes a complex sentence using JO 3 and 4. At first by using JO 4, two SSs, ‘You leave at [dTime]’ and ‘I leave at [dTime],’ are combined to form one complex sentence, ‘We leave at [dTime].’ JO 3 will combine this sentence and ‘We leave [dLocation],’ finally deriving a complex sentence like: ‘We leave [dLocation] at [dTime].’

A variety of sentences can appear from SPTs when genetic programming is applied. Figure 3 and 4 show an idea of how crossover and mutation operations are executed in SPTs. Figure 3 (a) and (b) show two SPTs before crossover operation and the shaded nodes are nodes that are going to be changed by crossover operation. Figure 3 (c) and (d) show the resulting SPTs after crossover. As Figure 3 shows, two complex sentences, ‘You are going to [aLocation] and you leave [dLocation] at [dTime]’ and ‘You leave at [dTime] and arrive at [aTime] by [tGrade]’ are changed to ‘You are going to [aLocation] and you leave at [dTime] and arrive at [aTime]’ and ‘You leave [dLocation] at [dTime] by [tGrade].’ Figure 4 (a) and (c) show the same SPT with a different shaded node before mutation, where the shaded nodes will be changed by mutation. Figure 4 (b) and (d) show the resulting SPTs after mutation. The upper example of Figure 4 illustrates the result of changing JO, and the lower one shows the result of changing CA. As shown above, the sentence ‘You leave [dLocation] at [dTime] and take the [tGrade]’ is changed to ‘You leave [dLocation] at [dTime] by [tGrade]’ by changing joint operator. The lower SPT is changed to be interpreted as ‘You leave [dLocation] at [dTime] and are going to [aLocation]’ by changing CA. With crossover and mutation operations, diverse SPTs might be generated to construct various sentences.

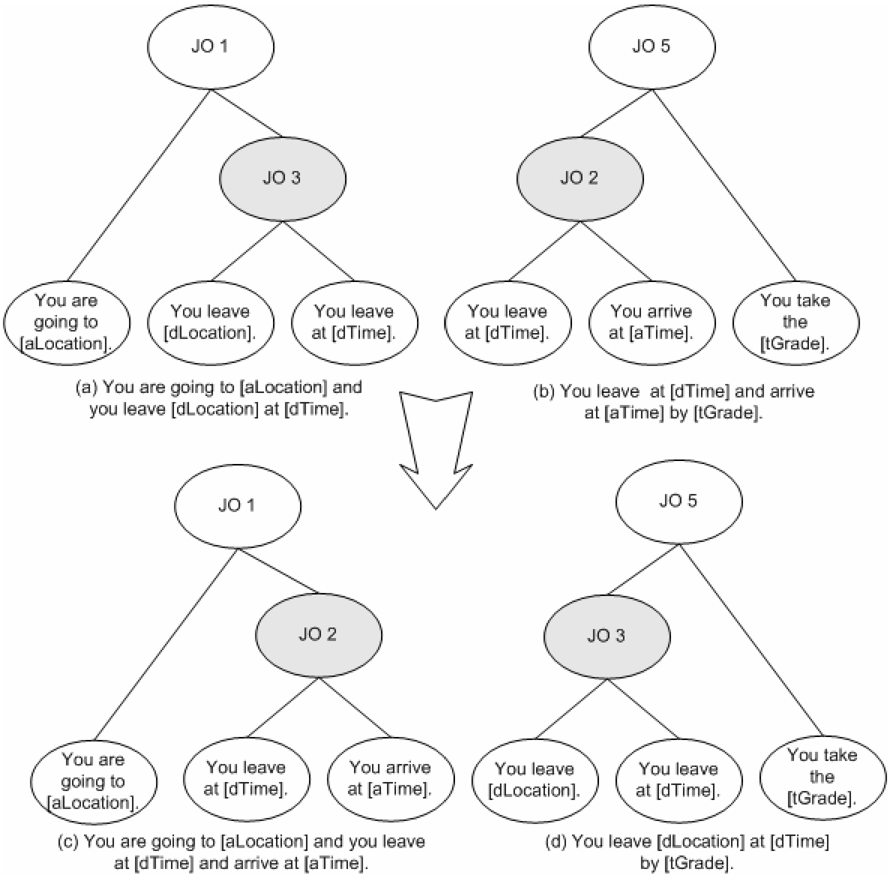

Figure 3. Crossover operation

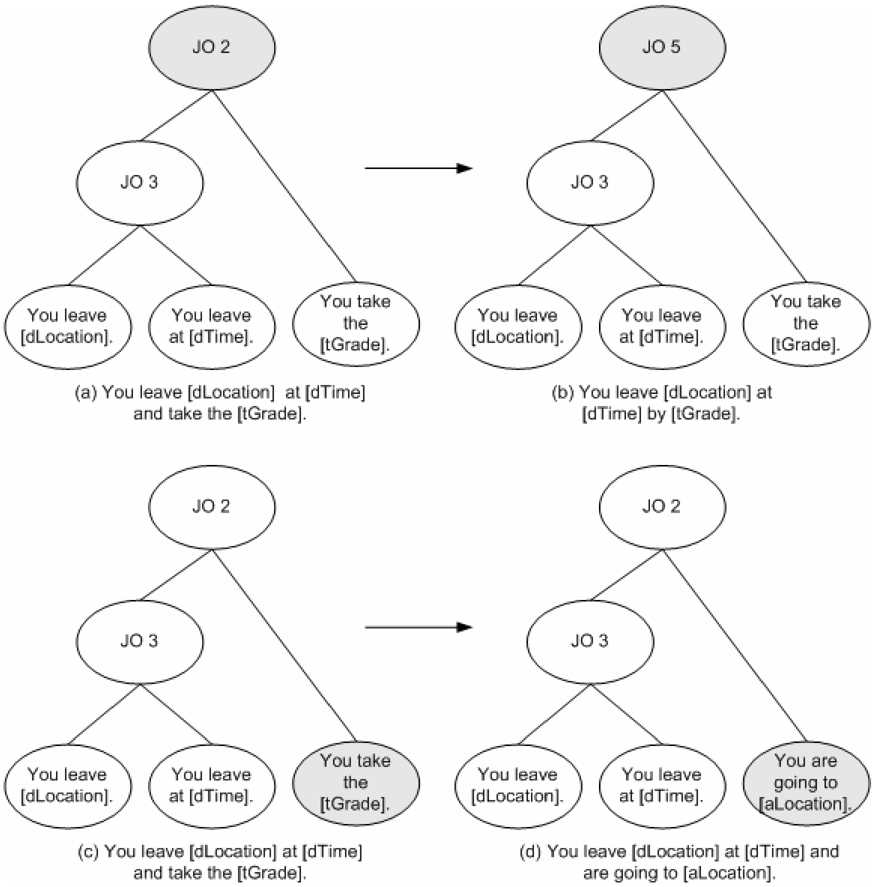

Figure 4. Mutation operation

3.2 Joint Operator

In this paper, we define JOs by analyzing Korean language. There are 5 operators which are applied differently for each of the 3 cases of combining two sentences: 2 statements, a statement and a question, and 2 questions. JOs are defined as below:

Let two sentences A and B:

A: subject (s1) + CA (c1) + verb (v1).

B: subject (s2) + CA (c2) + verb (v2).

- **JO 1**: Combine two sentences of A and B by using conjunction ‘and’. The result is ‘s1 c1 v1 and s2 c2 v2.’
- **JO 2**: Combine two sentences which have the same subject (s1 = s2). The result is ‘s1 c1 v1 and c2 v2.’
- **JO 3**: Combine two sentences which have the same subject (s1 = s2) and the same verb (v1 = v2). The result is ‘s1 c1 c2 v1.’
- **JO 4**: Combine two sentences which have the same communitive act (c1 = c2) and the same verb (v1 = v2) by making a proper subject (s3). The result is ‘s3 c1 v1’ where s3 is the subject which includes s1 and s2.
- **JO 5**: Combine two sentences which have the same subject (s1 = s2) and diferent verbs but can be replaced by a verb v3 which includes the meaning of v1 and v2. The result is ‘s1 c1 c2 v3.’

Table 1~3 show the examples of combining two SSs with those rules. Since the sentences are translated from the Korean, some sentences might not be matched exactly with JOs in English.

Table 1. Combining two statements

JO	Sentence 1	Sentence 2
	Result Sentence	
JO 1	You go to [aLocation].	He goes to [aLocation].
	You go to [dLocation] and he go to [dLocation].	
JO 2	You like [something1].	You hate [something2].
	You like [something1] and hate [something2].	
JO 3	You leave [dLocation].	You leave at [dTime].
	You leave [dLocation] at [dTime].	
JO 4	He is going to [aLocation].	She is going to [aLocation].
	They are going to [aLocation].	
JO 5	You leave [dLocation].	You are going to [aLocation].
	You are traveling from [dLocation] to [aLocation].	

Table 2. Combining a statement and a question

JO	Sentence 1	Sentence 2
	Result Sentence	
JO 1	You go to [aLocation].	Where does he go?
	You go to [dLocation] and where does he go?	
JO 2	You like [something1].	What do you hate?
	You like [something1] and hate what?	
JO 3	You leave [dLocation].	What time do you leave?
	What time do You leave [dLocation]?	
JO 4	Incorrect sentence	
JO 5	You leave [dLocation].	Where are you going?
	Where are you traveling from [dLocation] to?	

Table 3. Combining two questions

JO	Sentence 1	Sentence 2
	Result Sentence	
JO 1	Where do you go?	Where does he go?
	Where do you go? And where does he go?	
JO 2	What do you like?	What do you hate?
	What do you like and hate?	
JO 3	Where do you leave?	What time do you leave?
	Where and what time do you leave?	
JO 4	Where is he is going to?	Where is she is going to?
	Where are ther going to?	
JO 5	Where do you leave?	Where are you going to?
	Where are you traveling from and to?	

4. The Travel Planning Agent

This section introduces a travel planning agent implemented the proposed method, and shows how the evolution of SPTs works. The travel planning agent makes some conversations with a user to automatically collect the user's required information needed to decide which train to search.

When doing the same job, former "canned-script" systems tended to offer only simple and monotonous conversations, limited by their pool of available expressions and sentences. The presented system makes up a more dynamic and familiar conversations by coming up with a variety of sentences and expressions created in real-time. Figure 5 shows the structure of the train planning agent.

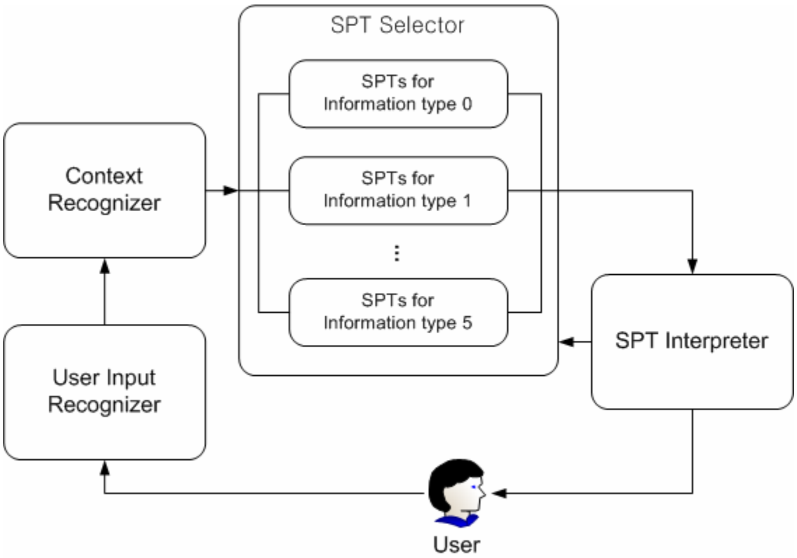

Figure 5. Structure of the travel planning agent

The context recognizer chooses a SPT group most suitable for creating relevant queries by recognizing the amount of knowledge available. Then the SPT selector randomly chooses a SPT from the given group. Then the selected tree is translated into a sentence by the SPT interpreter, finally being delivered to the user. Lastly, when the user's response has been given, the user input recognizer acquires available knowledge from the response. When enough information is gathered to determine a specific train, the agent will show corresponding trains.

4.1 Context Recognizer

The context recognizer relies on the amount of information available, namely the following 6 kinds: the user's place of departure, destination, departing date, departing time, arriving time and the seat class, to select a proper SPT group which will be generating a query sentence. In this paper, SPTs are classified into 6 groups where each group represents the number of information the agent has.

4.2 SPT Selector

After the context recognizer chooses a SPT group to produce the query, the SPT selector selects a random SPT among the chosen group, and passes it to the SPT interpreter.

4.3 SPT Interpreter

The SPT interpreter derives a query sentence from a SPT. Table 4 shows examples of SSs which are contained in the leaf nodes of a SPT. We stated above that leaf nodes of a SPT contain the corresponding SS. When there are no restrictions on choosing a SS to print, there are chances that a statement with unknown information tokens to be chosen. Awkward query generations arise again when a tree contains statements and questions together, which refers to the same kind of information. In this paper, this problem is solved by only indicating the kind of information involved in each sentence in the corresponding leaf nodes. The agent will automatically select a proper type of sentence (statement/question) consulting whether the agent knows the relevant information or not. Figure 6 shows the SPT which has domain knowledge.

Table 4. SSs used by a travel planning agent

User Information		Simple Sentence
Departure Location	Statement	Where do you leave?
	Question	You leave [dLocation].
Arrival Location	Statement	Where are you going?
	Question	You are going to [aLocation].
Departure Date	Statement	When do you leave?
	Question	You are leave on [Date].
Departure Time	Statement	What time do you leave?
	Question	You leave at [dTime].
Arrival Time	Statement	What time do you want to arrive?
	Question	You arrive at [aTime].
Seat Grade	Statement	What kinds of seat do you want?
	Question	You take [tGrade].

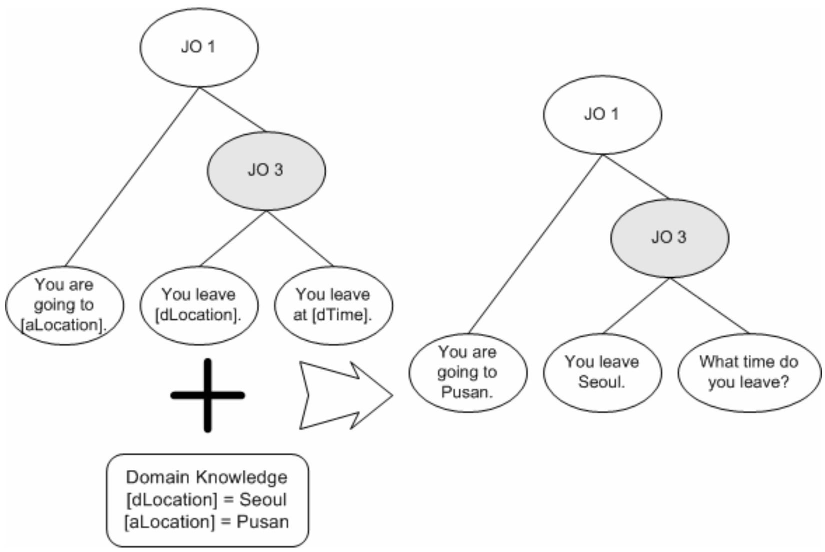

Figure 6. The SPT with domain knowledge

4.4 User Input Recognizer

The user input recognizer accepts the user's response to the offered query, and extracts relevant information by applying pattern matching and templates. That is, the agent performs a series of pattern matching with prepared templates to find the most similar template, and extracts relevant information from the template. For instance, when the user's response was "I am traveling from Seoul to Pusan", the most similar template will be "I am traveling from [dLocation] to [aLocation]" and the agent will be able to extract the user's place of departure, Seoul and the destination, Pusan

4.5 Interactive Evolution

SPTs are grouped by the criteria stated in Section 4.1. Each SPT group shows all available information of agent to the evaluator, and the evaluator will give fitness scores (between 0 and 10) to each of the query that the SPTs generated. The fitness is evaluated in regared of naturality of the queries. Then the evaluated trees evolve to their next generation. Finally, the system converges into the preference of the evaluator.

5. Experimental Results

5.1 Evolution Test

Ten subjects iteratively interact with the system in 90 generations. Crossover rate and mutation rate are set as 0.6 and 0.2, respectively. Each generation, the subjects evaluate fitness of all individuals. It needs much effort , so we set the population size as 20.

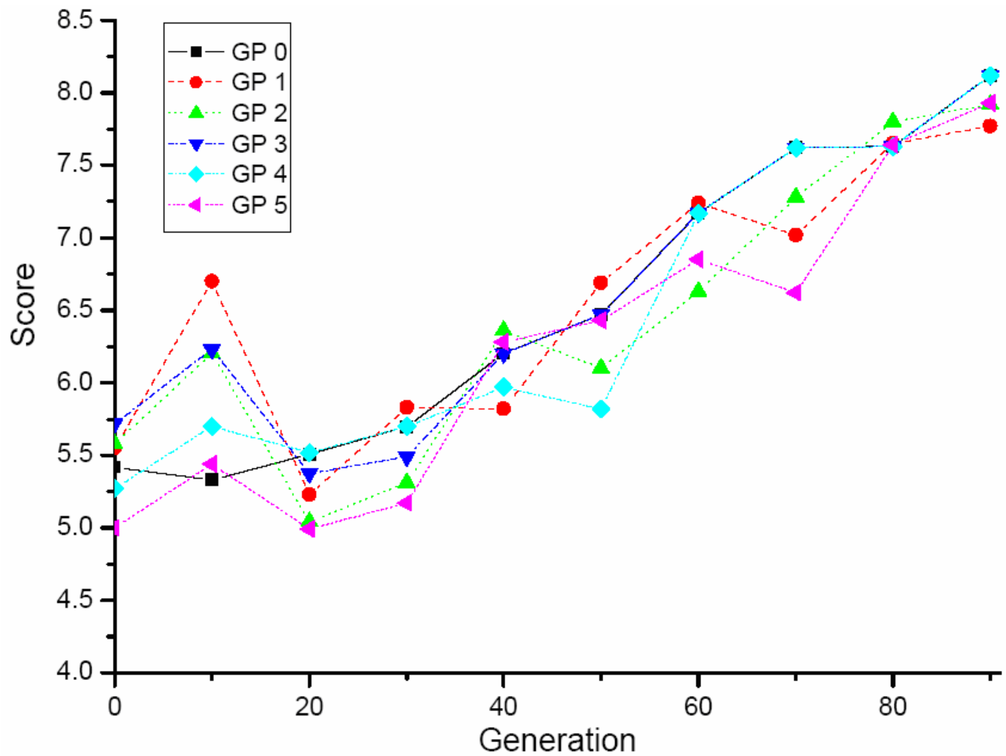

Figure 7. Score distribution through evolution

Figure 7 represents the changes of average score as growing generation. GP n means the fitness of STPs for information type n in figure 5. We limit the generation until 90 steps because more steps are the cause of overfitting and decreasing the diversity.

In the result, the score increases rapidly during the first 10 generations and decreases contrarily between 10th and 20th generation and increses continuously after 20th generation. This trend is occurred because the subjects felt at first steps of evolution that the unbalanced scentences change to more naturaly, so they evaluate relatily high. And they felt at middle steps that the changes do not get at the point of their expectation comparing with the first steps, so they evaluate low.

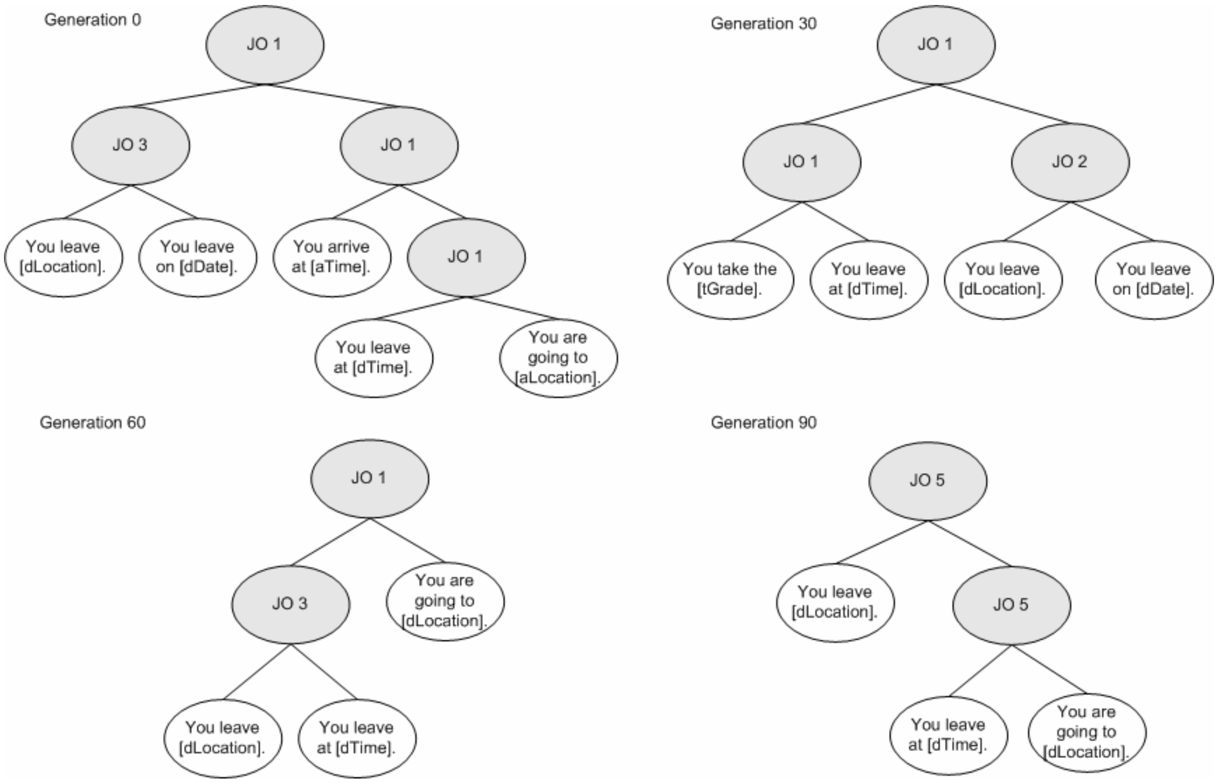

Figure 8. Sentences obtained by the system

Generation 0: You leave Seoul on 10th October and what time do you want to arrive? And what time do you leave? And you are going to Pusan.

Generation 30: What kinds of train do you want to take and what time do you want to leave? And you leave Seoul and leave on 10th October.

Generation 60: What time do you leave Seoul? And you are going to Pusan.

Generation 90: What time do you leave for traveling from Seoul to Pusan?

Figure 8 shows sentences obtained by the system, especially on the case of the third SPT while the system collects information on the departure location, the arrival location, and the departure data. As shown in Figure 8, the sentence in the initial stage was a little disorderly and complex while the sentence obtained through evolution became refined.

5.2 *Example of Dialogue*

Figure 9 shows an example of dialogue in which the user makes a travel plan assisted by the system. Each SPT works in the corresponding situation. In Figure 9, the information means that the domain knowledge which the agent has and the extracted information represent the extracted domain knowledges using pattern matching and template.

System:	Where do you travel from and to? Information: None. Use GP 0
User:	I travel from Seoul to Pusan. Extracted information: Departure location . Seoul Arrival location . Pusan
System:	When do you want to leave? Information: Departure location . Seoul

	Arrival location . Pusan Use GP 2
User:	10th October, 2004. Extracted information: Departure date . 10th October, 2004
System:	What time do you leave for traveling from Seoul to Pusan? Information: Departure location . Seoul Arrival location . Pusan Departure date: 10th October, 2004 Use GP 3
User:	About 10:00 PM. Extracted information: Departure time: 10:00 PM
System:	What kind of train do you want to take? Information: Departure location . Seoul Arrival location . Pusan Departure date: 10th October, 2004 Departure time: 10:00 PM Use GP 4
User:	I want KTX. Extracted information: Train kind: KTX
Sysetm:	There are two trains from Seoul to Pusan at 09:50 PM and 10:20 PM on 10th October, 2004. Information: Departure location . Seoul Arrival location . Pusan Departure date: 10th October, 2004 Departure time: 10:00 PM Train kind: KTX Trains selected.

Figure 9. An example dialogue

6. Conclusion

Generating sentences in the interaction with humans increases the reality and flexibility of a dialogue system. Moreover, a trainable system might adapt itself to various domains or users. Contrary to conventional methods that rely on the developer, the proposed method uses the domain knowledge as little as possible. Especially, genetic programming is used to generate domain-adaptive sentences dynamically. The experiments have shown the usefulness of the proposed method.

Although the proposal method is useful, there are some limitations. First, the process of evolution needs much efforts. As explained above, the evaluators must do the number of population times evaluations to evolve one generation. Second, if the method terminates evolution and it is applied to the system, there is no new sentences generated because there is no more evolution processes. As the future work, we will expand the join operator to generate further diverse sentences, and multi-sided experiments will be conducted to verify the scalability and usability of the proposed method. And to improve the usefulness of the system, we will apply some methods for easy evaluation and on-line evolution.

Acknowledgements

This work was supported by Brain Science and Engineering Reseach Program sponsored by Korea Ministry of Science and Technology.

References

Adorni G. & Zock M. (1996), *Trends in natural language generation: An artificial intelligence perspective*, Berlin, Heidelberg: Springer.

Bulyko I. & Ostendorf M. (2002), *Efficient integrated response generation from multiple targets using weighted finite state transducers*, Computer Speech and Language 16(3-4), 533-550.

Giuseppe R. & Allen L. (2000), *A stochastic model of humanmachine interaction for learning dialog strategies*, IEEE Trans. Speech and Audio Processing 8(1), 11-23.

Koza J. (1994), *Genetic programming: Automatic discovery of reusable programs*, The MIT Press.

McKeown K. (1986), *Language generation: Applications, issues, and approaches*, Proc. IEEE 74(7), 905-919.

Oh H. & Rudnicky I. (2002), *Stochastic natural language generation for spoken dialog systems*, Computer Speech and Language 16(3-4), 387-407.

Ratnaparkhi A. (2002), *Trainable approaches to surface natural language generation and their application to conversational dialog systems*, Computer Speech and Language 16(3-4), 435-455.

Seneff S. & Polifroni J. (2000), *Formal and natural language generation in the Mercury conversational system*, Proc. ICSLP, vol. 2, 767-770.

Theune M. (2003), *Natural language generation for dialogue: System survey*, TR-CTIT-03-22.

Walker M. A., Rambow O. C. & Rogati M. (2002), *Training a sentence planner for spoken dialogue using boosting*, Computer Speech and Language 16(3-4), 409-433.

Weiwei W., Biqi L., Fang C. & Baozong Y. (1996), *A natural language generation system based on dynamic knowledge base*, Proc. the 3rd Int. Conf. on ICSP, 765-768

Zue V. & Class J. (2000), *Conversational interfaces: Advances and challenges*, Proc. the IEEE 88(8), 1166-1180.