Grammar Representation Forms in Natural Language Interface for Robot Controlling

László Kovács and Péter Barabás

University of Miskolc, Department of Information Technology, Miskolc, Hungary
{kovacs,barabas}@iit.uni-miskolc.hu

Abstract. Grammar parsing and transformation of the incoming sentence into a corresponding semantic model are the kernel modules in natural language interface engines. This paper presents an analysis of the candidate grammar representation forms. The proposed grammar formalism is a combination of regular grammar and dependency grammar extended with predicate oriented annotations. The paper presents an implemented natural language interface for controlling humanoid robots in the environment of Hungarian language and similar languages.

Keywords: grammar models, human machine interface, natural language processing.

1 Introduction

The most important and flexible interface channel between humans and computers is the natural language. The role of the natural language interface (NLI) is steadily increasing in intelligent interface systems. The input source can be either speech or textual data that are converted into corresponding machine commands. The task of the NLI engine is to accept user's command formulated in natural language sentences and convert these commands into low level program function calls. The first proposals [3] on implementation of NLI systems were published in the late 1970s. Nowadays, the web-based and mobile-based client applications [1] are the main markets for NLI engines. The NLI engine integrates different operational units to perform the text to command conversion. The engine contains the following main modules [2]:

- Text parser API unit;
- Morphologic analyzer unit;
- Semantic analyzer unit;
- Function mapping unit.

One of the most important related application areas is the implementation of natural language interfaces for robot controlling. This application area has a significant history and its importance is still increasing. Regarding the pioneer works, Sato and Hirai [4] can be mentioned who implemented a language-aided instruction interface for teleoperational control. The applied language model was based on the keyword

P. Sinčák et al. (eds.), *Emergent Trends in Robotics and Intelligent Systems*,
Advances in Intelligent Systems and Computing 316, DOI: 10.1007/978-3-319-10783-7_7

identification mechanism. Later, Torrance[5] developed a natural language interface for an indoor office-based mobile robot in navigation area. RHINO [6] is a robotic guide in a museum which can move and describe particular exhibits. It can only recognize simple phrases without supporting true dialogues. RHINO's ancestor was Polly, a vision-based robot which interaction mechanisms were more primitive. Jiro-2 [7] is a mobile office assistant which can convey information and guide people through an office environment. A frame-based speech dialog system is used by Jiro-2 and it communicates in Japanese. AESOP 3000 [8] is a surgical robot which is controlled by voice in heart surgery and does not provide a full dialog system. The implemented prototype systems mainly support only the dominating languages. The goal of our project was to implement an NLI module for the Hungarian language and to develop a flexible and efficient language processing model.

2 Grammar Representations

The meaning of sentences is partially encoded with syntax of a language. The rules of sentence generation are defined by grammar. The theoretical background of grammar analysis is the theory of formal grammar where a language is defined as a set of sentences:

$$L = \{s^*\}, s \in A \tag{1}$$

where A denotes an alphabet of a language. The elements of the alphabet are called terminal symbols. The grammar describes how to construct valid sentences from the alphabet. The main operators are the

concatenation (ab) ;
Kleene star operator (a*) .

The set of related complex structures generated by concatenation operations are usually encoded with non-terminal symbols. The symbol for valid sentence is denoted by S. The grammar consists of production rules in the form

$$\alpha \rightarrow \beta$$

where α,β are sequences of terminal and non-terminal symbols. The complexity of valid sentences depends on the complexity of the grammar rules. According to the Chomsky classification, the following main grammar classes are defined in the literature:

- regular grammar,
- context-free grammar,
- context sensitive grammar,
- unrestricted grammar.

The main shortcoming of formal grammar formalism is that it ignores the semantic roles of the terminal and non-terminal symbols. As in many applications, as in the

case of NLI engines, the management of the semantic context is a key element of the analysis, there are some extensions of the standard formal grammars to include the semantic aspects, too. The Word Grammar proposed by Hudson [9] belongs to this category. Word grammar represents a grammar with a knowledge graph including all four levels of human language, namely the semantic level, the syntax level, the morphology and the phonology.

A different approach is implemented in the Dependency Grammar proposed by Tesniere [10]. Dependency grammar uses dependency description between the words of a sentence. The dependency has a head (verbs) and some dependents. The dependency relation corresponds to grammatical functions. The dependency relationship is described with a dependency tree. A similar formalism is used in the Link Grammar. The link grammar uses only binary relationships, i.e. complex relationships within a sentence are represented with relationships between two words of the sentence.

Considering the standard grammar representations, another shortcoming is the dominance of the sequence-oriented structure. In many dominating languages, the ordering of the different parts of speech units has an important role in encoding the semantics. On the other hand, in many other languages, there is no dominating word order, many different orders are valid. To cope with the free-order approach, some extensions of the standard grammars are proposed.

One of the approaches of this domain is the FAwtl-grammar (Finite Acceptor with translucent letter)[11], where the base regular automaton is extended with some new features. A node in the automaton is assigned to a set of translucent letters. These letters are skipped during the parse operation. The automaton processes the first symbol which is not a member of the current translucency set. The processed symbol is then removed from the string and the head of the automaton jumps back to the beginning of the sentence. Using this technique, it is possible to model some order-free structures.

The conversion of the input word sequence into semantic structure is called function mapping. The goal of this step is to transform sentence analysis into a function description in order to be able to execute the real action related to the description. Two main tasks can be defined to satisfy the function mapping:

1. Find the proper function description to sentence analysis.
2. Map nodes (concepts) of sentence analysis to function parameters.

To be able to map the nodes of the sentence analysis tree into parameters, description should also be completed with some constituent information, as

$$p = (c, \{\kappa\}, f) \tag{2}$$

where c is the concept belonging to the parameter, { κ } is the set of constituents belonging to the parameter, f is the compulsory function which assigns mandatory value to the parameter. However, there is a problem which was yet not handled properly with the previous approaches: when more instances of the same concepts can be found in the analysis tree, their mapping is ambiguous.

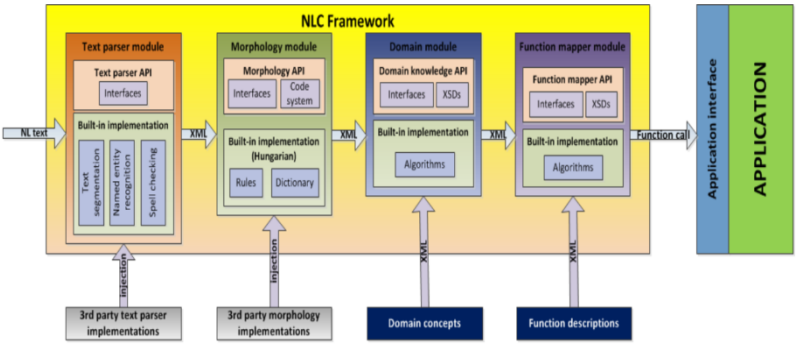

Fig. 1. Operational framework of the model

3 Efficiency of the Representation Models

The simplest solution is to use a keyword language where sentences consist of only one word. In this case, a simple regular grammar can be used. The cost of the parsing operation depends on the applied search mode, where exact or fuzzy search can be used. In the case of fuzzy matching, some differences are tolerated. Using an exact keyword matching and a corresponding index structure, the parsing requires only O (log(N)) cost, where N denotes the number of keywords in the lexicon. In the case of fuzzy matching, a more complex index structure is used, like the vantage-point structures [12]. In this case, a distance function should be defined to measure the similarity of the words. Usually, the edit distance is selected for dissimilarity measure. The calculation of the edit distance belongs to the O(m2) family, where m denotes the average length of the words.

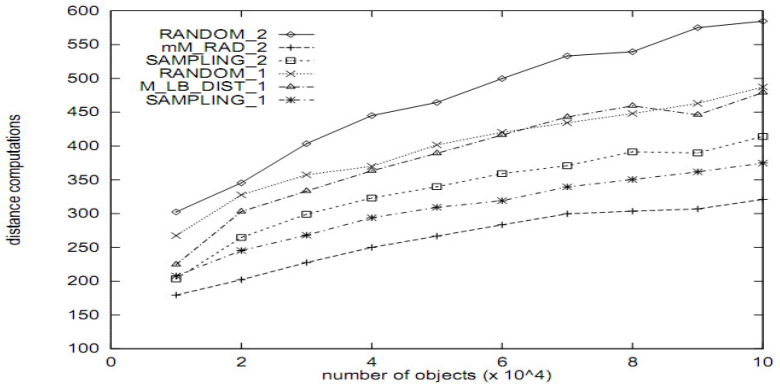

Fig. 2. Cost function for NN search (source: [13])

The cost of matching in the index tree is higher than in the case of exact matching; as several paths should be parsed to locate the nearest neighbor element, it can be estimated with O(logk(N)). A good cost analysis for the M-tree variant is given in [13].

The functionality of this approach is very limited, there is no way to submit additional parameters to the operation.

The next complexity level is the application of restricted natural language structure, i.e. only flat structures are allowed with simple POS components. This means that the sentence can be modeled with a star-like dependency graph. This model enables the passing of the required execution parameters and it is simple enough to meet the efficiency requirements. If a fixed order exists in the applied language, this language can be modeled with a regular grammar. Using an exact matching, the cost of parsing is only O(nlog(N)), where n denotes the average length. In the proposed prototype system, a free-order language variant was tested, thus a star-graph model was selected as a grammar representation. In this model, the central node is the predicate word which has some dependent POS components. It is assumed that every POS component has a simple linear structure and has a unique semantic role. Based on this simplification assumption, the parsing cost could be reduced to O(n2log(N)). As n is a small number, this representation model provides an efficient execution of the parsing process. The parsing process returns a distance value for the matching of an incoming sentence and a graph model. This distance value describes the degree of similarity. Enabling a fuzzy search for the predicate part, an NN-search operation is required, where the cost value will be equal to O(n2logk(N)).

4 Robot Control Application

In the case of Hungarian language, the speech recognition and the speaker-independent ASR are in their childhood and there is no exact, fast and accurate solution which could meet the requirements of real-time usage. There are some by-pass solutions with restricted functionality. Nuance has some promising products like Nuance Recognizer or Dragon which increasingly support Hungarian language. In sample application, the Nuance Recognizer is used to convert speech into written text. Well-defined grammar XML files should be built which describe the word sequences that the robot is able to recognize. The benefit of this representation is that the grammar can be arbitrarily recursive so the number of combinations can be increased. It comes for a certain price: the process of recognition will be slower. The speech recognizer or the keyboard generates the input of the NLC framework which is well-adapted with domain knowledge and function descriptions. The structure of application can be seen in Figure 1.

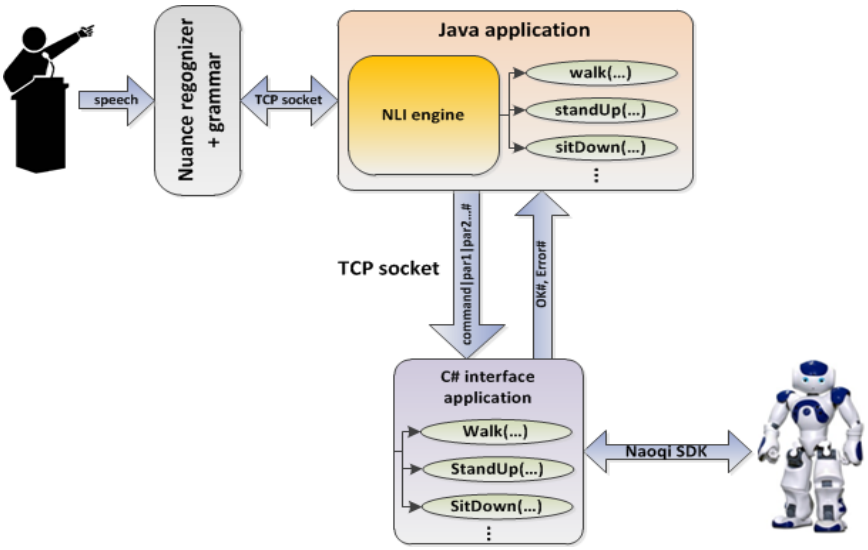

Fig. 3. The structure of Robot Controlling Application

The domain knowledge is encoded in a set of available functions. The NLI engine supports 18 standard functionalities of the robot, which includes, among others, the following elements:

- Robot can „close its eyes" which means hiding camera window in application;
- Robot can lay down from any position;
- Robot can walk specified length in any direction.

The applied Aldebaran's Nao[14] humanoid robot has a software development kit which provides an application programming interface in several languages like C++, Python, C#. At the time of development, there was no Java support yet. A by-pass solution had to be found to communicate with Nao from the natural language interface framework written in Java. The solution was to develop an interface application in C# which connects to the robot and calls its API methods. The C# application gets commands via socket communication where the interface application acts as a socket server and the NLC framework has a socket client part.

The socket messages have simple structure which consists of a command name and parameters. A sample socket message for the walk function is as follows:

```
Walk|0.4|forward#
```

This instructs the robot to walk 40 centimeters forward. The delimiter is 'l' between function names and parameters and the '#' sign is the terminate symbol of the message. The interface application calls the corresponding NaoQi API function and Nao makes the movement.

Based on the performed tests, the implemented architecture provides a suitable NLI framework with basic functionality. Regarding the execution-cost factor, the STT (speech to text) module is the dominating cost component. The generation of the command text required usually 1-2 seconds, while the location of the corresponding semantic graph took only 0.2 seconds. The knowledge base of the test system contains the following concepts:

> stand, sit, lay, sorry, wave, nod, exclaim, make excercises, see, close eyes, say hello, walk, turn, introduce, welcome, raise, lower, twist, look, body_part, limb, head, leg, arm, hand, elbow, direction, unit, person, side, forward, backward, left, right, back, up, down, here, around, meter, centimeter, degree, step, Ági, Peti, Szabi, Attila, left , one, right one.

The main drawback of the limited word-set was the lower accuracy factor. The recognized language contains only a small subset of real-life domain, thus, for human commands containing untrained words, the user should reformulate the query command. For the query refinement step, the architecture contains a dialog module. If a command has missing parameters or the application does not understand a command, a question entry will be shown in the conversation panel.

References

1. Lee, G., Lee, J.H., Rho, H., Park, Y.T., Choi, J., Seo, J.: Interactive NLI agent for multiagent Web search model. In: 4th World Congress on Expert Systems (1998)
2. Barabás, P.: Domain- and Language-adaptable Natural Language Controlling Framework, PhD dissertation, University of Miskolc (2013)
3. Sondheimer, N.K.: Spatial Reference and Natural Language Machine Control. Int. Journal of Man-Machine Studies, 329–336 (1976)
4. Sato, T., Hirai, S.: Language-Aided Robotic Teleoperation System (LARTS) for Advanced Teleoperation. IEEE Journal on Robotics and Automation, 476–480 (1987)
5. Torrance, M.C.: Natural Communication with Robots. Master's thesis. MIT, Department of Electrical Engineering and Computer Science, Cambridge, MA (1994)
6. Burgard, W., Cremers, A.B., Fox, D., Hahnel, D., Lakemeyer, G., Schulz, D., Steiner, W., Thrun, S.: The interactive museum tour-guide robot. In: Proceedings of the Fifteenth National Conference on Artificial Intelligence, Madison, Wi (1998)
7. Asoh, H., Matsui, T., Fry, J., Asano, F., Hayamizu, S.: A spoken dialog system for a mobile office robot. In: Proc. of Eurospeech 1999, pp. 1139–1142 (1999)
8. Versweyveld, L.: March: Voice-controlled surgical robot ready to assist in minimally invasive heart surgery. Virtual Medical Worlds Monthly (1998)
9. Hudson, R.: Language Networks: The new Word Grammar. Oxford University Press (2007)
10. Tesniere, L.: Elements de syntaxe structurale. Paris, Klincksieck (1959)
11. Nagy, B., Kovács, L.: Linguistic Applications of Finite Automata with Translucent Letters. In: ICAART 2013: 5th International Conference on Agents and Artificial Intelligence, Barcelona, vol. 1, pp. 461–469 (2013)

12. Kovács, L.: Reduction of Distance Computations in Selection of Pivot Elements for Balanced GHT Structure. Transaction on Machine Learning and Data Mining 6, 14–37
13. Ciaccia, P., Patella, M., Rabitti, F., Zezula, P.: Indexing Metric Spaces with M-tree. In: Sistemi Evoluti per Basi di Dati SEBD, pp. 67–86 (1997)
14. Aldebaran: NAO. Aldebaran, http://www.aldebaran-robotics.com (retrieved June 6, 2013)