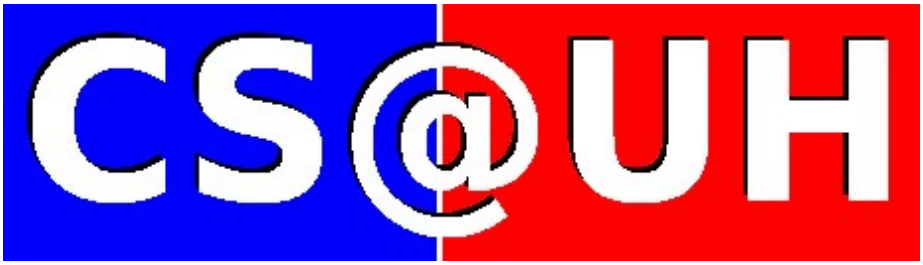

AN OBJECT-ORIENTED APPROACH IN REPRESENTING AND PARSING THE ENGLISH GRAMMAR

W. Faris and K. Cheng

Department of Computer Science
University of Houston
Houston, TX, 77204, USA
http://www.cs.uh.edu

UH-CS-08-03

February 28, 2008

Keywords: Communication, Natural Language Processing, Artificial Intelligence, Parsing, Grammar.

Abstract

Communication using a natural language is an important requirement for any artificial intelligence program. This paper first defines clearly the purpose of the communication problem, and presents its three sub-problems. The first sub-problem is to learn the grammar of the natural language English in order to teach to and store it effectively within the program. The second sub-problem is to parse a given sentence and translate it into a thought for the program to understand and carry out the intention of the sentence. The third sub-problem is to use the learned natural language to produce a valid sentence given an internal thought. In this paper, we describe what we use in our program to represent a subset of the English grammar and our initial effort in implementing the parsing of an English sentence. Object-Oriented paradigm is used to analyze the problem and design the solution to attack the communication problem. It provides an important first step in achieving intelligent communication between a computer and human.

AN OBJECT-ORIENTED APPROACH IN REPRESENTING AND PARSING THE ENGLIGH GRAMMAR

W. Faris and K. Cheng

Abstract

Communication using a natural language is an important requirement for any artificial intelligence program. This paper first defines clearly the purpose of the communication problem, and presents its three sub-problems. The first sub-problem is to learn the grammar of the natural language English in order to teach to and store it effectively within the program. The second sub-problem is to parse a given sentence and translate it into a thought for the program to understand and carry out the intention of the sentence. The third sub-problem is to use the learned natural language to produce a valid sentence given an internal thought. In this paper, we describe what we use in our program to represent a subset of the English grammar and our initial effort in implementing the parsing of an English sentence. Object-Oriented paradigm is used to analyze the problem and design the solution to attack the communication problem. It provides an important first step in achieving intelligent communication between a computer and human.

Index Terms

Communication, Natural Language Processing, Artificial Intelligence, Parsing, Grammar

I. Introduction

In an artificial intelligence program, the capability of the program to communicate with the external world allows the program to gain knowledge, interact with its environment, and to express what it knows. An important form of communication is a natural language; a language people use to communicate with each other. In a multi-agent artificial intelligence program [1], this should be the duty of the communication agent. The communication agent should be responsible for learning the grammar of the natural language, and store it in the memory agent. It should also use this grammar to understand thoughts expressed to it, and to produce responses to its user. Similar to all other knowledge learned by the program, the knowledge on how to communicate should be taught instead of pre-programmed into the program. This paper will focus on the problem of representing a grammar and parsing a given sentence. The component knowledge required to specify the grammar of the natural language English is stored in the memory agent. Parsing, an algorithm of the communication agent, is an important first step in understanding a sentence.

We have proposed *A Learning Program System* (ALPS) [2] with the goal of learning the knowledge that a human is capable of learning. The focus has been on the development of the memory agent to store knowledge and their relationships. Basic capabilities, such as creating a new category, adding objects, attributes, and properties of a category have been provided. We have recently developed two major knowledge components of categories: hierarchy [3] and definition [4]. Hierarchy allows both generalization and containment relationships among categories to be specified and stored. Definition specifies the necessary and sufficient condition of a specific category that may be used to classify objects. Currently, all these learning capabilities are accomplished through special interface objects that request the appropriate knowledge to build the more complicated knowledge. However, this will require us to continue to develop these special interface objects for each new kind of knowledge that the program needs to learn. What we want to accomplish is to use a simple English sentence instead, so that this single interface can be used to learn most knowledge. This paper will describe our current effort in creating the communication agent to learn a subset of a natural language, specifically English. This subset will be used to

understand sentences issued by a human to access the capabilities of the current program. It may be used to declare knowledge and to answer questions. For example, the sentence 'John is a human.' will add the knowledge object John to the human category. On the other hand, the question 'What is John?' will require the program to find and provide its answer.

Our program is implemented in C++, and uses Object-Oriented paradigm in its analysis and design. An advantage of an Object-Oriented solution is the reusability of the implementation. We have reused the hierarchy class to implement the relationships among the different kinds of sentences, and grammar terms such as personal pronoun is a pronoun, and pronoun is a noun. Object-Oriented solutions also allow mix-and-match of the different components, and provide simpler future extensions of any component. We extended the condition class of the definition project by adding three sub-classes: the pattern, fulfill, and the sequence conditions.

The rest of this section is a brief review of previous research on the different formal models of grammar. Section 2 presents the communication problem and its three sub-problems along with our approach on teaching the system. Section 3 presents the details on how we represent the English grammar in our program. The representation contains multiple components needed to define the different grammar terms of the natural language, English. In Section 4, we describe a distributed algorithm for parsing a given sentence presented to the program. Section 5 concludes the paper along with a look at future research on the communication problem.

Research in natural language processing has attracted a lot of attention leading to a variety of different formal models of grammar. Besides context free, this includes grammars such as categorical [5], Montague [6], combinatorial categorical [7], and type-theoretic [8]. Functional programming languages such as Miranda [9] and Haskell [10] have been proposed and several parsers for functional programming languages [11-13] have been developed. Many natural language interface systems such as Lolita [14], and Windsor [15] have been built using these parsers. A comprehensive survey of the different grammars including the lazy functional programming approach can be found in [16]. One major problem on using functional programming approach is that additions to accommodate a complex aspect of the natural language may require a major change to many existing structures [17]. Furthermore, although the intention of lazy functional programming is to avoid the duplication of computation, it is quite possible to have unforeseen side effects.

II. The communication problem

The goal of learning a natural language is to teach the system to use this language to communicate. Following the instruction issued by its user, the system is assumed to have a certain amount of capability to perform the task. It is also assumed to have the intention to try to express its thoughts and/or belief. After learning the natural language, the communication capability of the program will not be limited to some special input/output interface objects. Instead, the system will possess the ability to communicate in that natural language. When presented with an instance sentence of the natural language, the program is now able to understand what the user intends. The program should also have the capability to express its internal thoughts to the external world using that natural language. In other words, the communication problem is about learning the agreeable ways used for the exchange of ideas. The responsibility of learning a natural language is subdivided into three sub-problems. The first of which learns the agreeable ways of communication, which is the grammar of the natural language. The second sub-problem uses the learned grammar to understand a given instance created by the grammar. Finally, the third sub-problem uses the learned grammar to present an internal intention. In order to solve the second and third sub-problems, the learning of the grammar cannot be limited to learning the simple grammatical formats and rules. More importantly, it also includes learning the intention of each grammar term, which we refer to as the role of the grammar term.

We use an English sentence to express a complete thought. Understanding and expressing multiple thoughts, such as complex sentences or paragraphs, will not be the concerns of the current paper. Our investigation will focus on learning a subset of the grammar rules for present tense sentences presented in an active voice. Sentences using other tenses and passive voices will be added in the future. As for the problem of subject-verb agreement, we will use stemming to change plurals into singular spelling words. Stemming is a common technique to transform all variations of a word to a common root word. Words like 'computers', 'buses', 'mice', and 'runs' are translated to 'computer', 'bus', 'mouse', and 'run', respectively. The original form of a word is retained and referenced in subsequent processing. Stemming will reduce the time required of parsing a sentence. Currently, we depend on a variation of the word banks provided by WordNet to carry out the stemming process. WordNet is a lexical

database for the English languages developed at The Cognitive Science Laboratory at Princeton University (http://wordnet.princeton.edu/).

Our approach in learning a natural language follows closely on how a human learns a language; namely, that the grammatical structures are taught by teachers. Learning a language is a long and intensive process that a student cannot accomplish in one sitting. The process is iterative such that simple grammar structures are first learned and used, and then learning alternatives that are more complicated later. Although the initial set of grammar taught to the program in this paper will be a limited subset of English, the framework that we produced will allow our program to handle sentences that are more intricate after more grammar has been taught. This is the first step in achieving intelligent communication between the computer and its human users. In addition, when organization skills were taught later in reading and writing, our program will be able to understand multiple sentences in a paragraph, and to present a theme with several related thoughts in an organized manner. Since natural language is complex and may not conform to any formal grammar, we are not concerned with a specific formal grammar and any improvements on it to accommodate a natural language. Our solution will not be limited by any specific formal grammar, but rather, depends on the presence of good teachers who will teach the correct usage of the grammar terms. The teacher simply provides the purpose of each grammar term and the program will use this information to link the communication capability to the internal capability. In other words, based on the teaching, the intention of a given sentence can be translated into an internal action that manipulates knowledge in or adds knowledge to the knowledge base. In response, the intention of the program to express a thought can be translated into a meaningful sentence based on these same teachings. Our responsibility is to provide basic features that allow the system to learn these grammar rules. We have developed a program flexible enough to accommodate variations in teaching styles and additions to existing teachings. This is in stark contrast from the functional programming approach in which new rules will require a major overhaul of previously defined structures.

III. The Grammar representation problem

The first sub-problem in learning the grammar of English requires us to first design the representation of all the different knowledge components of the grammar. Learning English grammar involves learning the various grammar terms in English: such as sentence, complete subject, verb, and preposition. Based on the understanding of the requirement of using the English grammar, there are four major components for each grammar term: structure, role, kind, and rule. Not all components are required for every grammar term, and we can define a specific grammar term in any combination of these components. The structure of a grammar term defines exactly what composes it. It is used in identifying the different parts of a sentence during parsing and in generating a sentence to reflect a specific thought of the learning program. The role of a grammar term defines the purpose of the term, allowing the program to understand the exact intention of the speaker. These roles bridge the communication agent to the rest of the program. A term can have multiple kinds, which are considered subsets that may share the same structure but must have different roles. Finally, a rule specifies the condition that must be satisfied. Rules can be applied directly to the grammar term or to one of its structures.

The structure of a grammar term may be either a sequence or an alternative. A sequence specifies the order of multiple grammar terms used to construct the entire grammar term. For instance, a simple structure of a sentence is the sequence of a complete subject followed by a complete predicate and ending with a form of punctuation. In this case, complete subject, complete predicate, and punctuation are grammar terms used to define the sentence grammar term. Currently, a term's occurrence in a sequence may be compulsory, optional, zero-to-many, or implied. By definition, in a verb phrase, the main verb is necessary and thus its occurrence is compulsory, but a verb phrase may also have several optional helping verbs. The implied option is currently used in a structure for second-person imperative sentences, where the subject of 'you' is not always given in a sentence such as 'Go to the store'. In other words, 'you' is implied as the subject. It should be noted that if a grammar term is a sequence, then it is quite possible that additional sequences may define it; however, it can only be satisfied by one such sequence. An alternative specifies the different possibilities that a term can assume. The possibilities may be other grammar terms, existing knowledge, or special words that the current system does not have a knowledge object associated with them such as 'you', 'of', and, 'how'. They are implemented as several sub-classes of alternatives. The first sub-class is for alternative grammar terms. For example, the grammar term main verb is an alternative comprised of an action verb, a linking verb, or a possessive verb. The next sub-class is for alternative kinds of knowledge. This sub-class creates a connection between the communication capability and the existing knowledge base, which is also acquired dynamically by the learning program. For example, the grammar term noun is currently taught as

an alternative of knowledge kinds: category, object, concept, and data type. These alternatives are responsible for finding the knowledge object that the program has already learned, such as human, john, force, and fractions, respectively. Two more sub-classes of alternatives deal with words that have exact spelling, such as prepositions and personal pronouns. In one class, the words are stored without any additional information such as prepositions. However, the other sub-class uses a multi-dimensional table format, as is the case with personal pronouns. The reason is that it will facilitate the process of deciding which pronoun to use when the program is trying to formulate a sentence. Details of the data structure to implement this class can be found in [18].

The role of a grammar term defines the purpose of the term. These roles will take a successfully parsed sentence and guide the program in the proper way to match the user's intentions. A role can define the intention of the entire sentence, guide the program towards the proper internal commands to execute, or label a grammar term to help in its identification. We have mentioned that the intent of a sentence is to express a thought, thus, this is considered its role. However, there are different types of thoughts and thus different types of roles for a sentence: specifically a statement, a question, a command, or an exclamation. All of which can be used to describe the intent of a particular sentence. When it comes to guiding the program, examine, as an illustration, the following sentences that use a form-of-be linking verb: 'John is a doctor' and 'Humans are mammals'. In both cases, the verb defines the subject as the predicate. For this reason, the role of the forms-of-be grammar term is to define the subject. Thus, when encountered with such a sentence, the program will be able to create a relationship between the subject and the predicate. John will be defined as a doctor and humans will be classified as mammals. Furthermore, the role of a grammar term may be determined by its position in a sentence. For instance, a noun phrase is labeled as a subject, direct object, or indirect object depending on where it appears in a sentence and this label helps to explain its role. In this case, the role is associated with the term as it appears in the sentence structure, not with the grammar term noun phrase itself. Finally, a term may be given an option of several roles, which it will not be assigned until a rule has been verified. As we will see in the next paragraph, a rule can either chose a role for a grammar term, or restrict it from obtaining one. All these roles are part of the grammar that the user needs to teach to the system.

The kind of a grammar term is a sub-category of that term, and it is possible for a category to have its own sub-kinds. A term with different kinds is similar to one with alternative grammar terms. However, each kind must be taught with an attached role, and its role will overwrite the role of its parental grammar term. As previously stated, the role of a sentence may be chosen from many possible roles. A declarative sentence kind is associated with the statement role, an interrogative sentence is paired with the question role, and a decision question, a sub-category of an interrogative sentence, is paired with the decision question role. In the next section, we will see how rules help to select which kind. When a kind of sentence is chosen, its associated role will supersede that of the thought role. Since all the knowledge that can be added to a kind is the same for a grammar term, each kind is simply implemented as another grammar term without the need to implement it as a separate class.

A rule specifies the condition that a fulfilled grammar term or its sub-term needs to satisfy. Given an English sentence, a grammar term is fulfilled when the content of its structure is determined. In the discussion of rules, the reference to a grammar term is understood to be a fulfilled grammar term. We implement rules using the condition class developed in the learning of definitions [4]. We added three subclasses of condition: pattern, fulfill, and sequence. The pattern condition verifies that a term matches a certain pattern like checking the ending punctuation of a sentence. The fulfill condition ensures a particular grammar term has been satisfied properly, such as the verb being a linking verb. This verb condition can be used to decide that the complement is a subject complement. The responsibility of the sequence condition is to assert a certain ordering of the grammar term in areas were an optional amount of alternatives are acceptable. For example, a sentence could have a series of helping verbs, however, there cannot be duplicate helping verbs of the same form and certain helping verbs must precede others.

There are two independent dimensions for the different types of rule. Each type of rule is used for different purposes and at a different time. The dimensions help us to identify the correct type of rules to use. The first dimension determines if the rule is applied to a grammar term's structure or to one of its kinds. The second dimension defines whether it is a restriction or a choice rule. A restriction rule is the necessary condition, while the choice rule specifies the sufficient condition, which needs to be satisfied, respectively. The resulting four combinations for rules are kind-restriction, term-restriction, kind-choice, and term-choice. A kind-restriction rule specifies a necessary condition that the kind must satisfy. For example, the kind of a given sentence should be a decision question if its structure is satisfied. The kind-restriction rule for a decision question states that its complete subject is not fulfilled by an interrogative pronoun. If this rule is satisfied, the sentence is confirmed as such a kind and consequently assigned the associated decision question role. A term-restriction rule defines the

condition that a grammar term must satisfy. For instance, when a complex structure of the grammar term ‘noun phrase' contains an article or a prepositional phrase, the term-restriction rule requires that the noun cannot be satisfied by a pronoun. A kind-choice rule defines the sufficient condition to decide the kind of the grammar term. Once the kind is chosen correctly, its associated role is recorded. For example, the basic structure of a sentence contains several kind-choice rules, one of which states that if the structure ends with a period then it is sufficient to conclude that the given instance is a declarative sentence. Finally, a term-choice rule defines the condition to fulfill a grammar term by one of its alternatives. It speeds up the parsing process by dictating which alternative grammar term should be applied, instead of trying them one by one. Take for instance, the complement of a sentence is composed of two alternatives, subject and object complements. The English grammar states that if an action verb fulfills the main verb, then the complement must be an object complement. Otherwise, a subject complement should fulfill the complement. Notice that both the rules and the roles may be attached to the structure or to the entire grammar term, so our learning interface has to provide the opportunity to introduce them as needed.

IV.A Simple, distributed parsing algorithm

Based on the understanding of the requirement of using the English grammar, we have completed the learning of a small subset of the English grammar, and have started working on the second sub-problem. The second sub-problem involves understanding a given sentence using the learned grammar. To understand a sentence, the learning program first parses the sentence, then executes accordingly and appropriately based on the current state of the system. Parsing determines the exact content of each grammar term and their respective roles, while ensuring all rules are satisfied. We have already implemented a simple algorithm for parsing. Our solution parses a given English sentence in a top down, depth-first order. During parsing, each grammar term will consume the part of the sentence satisfying that term, leaving the rest of the sentence to be parsed by subsequent terms. A grammar term uses its structure to parse, so each type of structure has its own parsing algorithm. As a result, the parsing algorithm is a distributed solution in nature.

For the alternative structure, its parsing algorithm will try the alternatives one by one, and the next alternative is tried only when the previous alternative fails. For instance, the structure for a noun includes class, concept, data type, and object in that order. When encountered with the word ‘John’, all alternatives fail until finally discovering that ‘John’ is an object. For the sequence structure, it will also parse each successive term in the given order. If a term is satisfied without violating any rule, its result is stored. However, when a term fails, the processing differs depending on its occurrence requirement. For an optional, zero-to-many, or implied term, the fulfillment of the term is considered completed. On the other hand, if a compulsory term fails, then either our algorithm backtracks or the parsing of this sequence fails. If the previously fulfilled term is not compulsory, our algorithm will backtrack. Notice that this may be the last part of a term that appears zero to many times, but not the whole term. However, if the most recently fulfilled term is also compulsory, then the parsing of this sequence fails. Finally, since a sequence structure is actually an alternative of multiple sequences, after one alternative sequence fails, our algorithm will try the next sequence. Our parsing algorithm applies each of the four different kinds of rules at a different time during the parsing. If term-choice rules exist for an alternative structure, our algorithm uses them to decide which alternative term to parse. Our algorithm will apply any term-restriction rule once a grammar term has been fulfilled successfully. After that, if kind-choice rules exist, the parsing algorithm will apply them to determine the exact kind of the grammar term and store its associated role in the parse result. For grammar terms that have no role chosen by the kind-choice rules but do have a generic role, this role will be assigned to the grammar term. Finally, our algorithm will check kind-restriction rule to make sure the necessary condition is satisfied by the chosen kind.

The following describes the main components of the subset of the English grammar that has been taught to the learning program. This includes the basic sentence structure for different kinds of sentences: declarative sentence, exclamatory sentence, interrogative sentence, and imperative sentence, which is a second-person command. In addition, the subset also contains the structures for decision question and third-person command. The general components of a sentence include complete subject, verb, and both subject and object complements. The complete subject and the complements are all noun phrases; a phrase focused around a noun and may include modifiers, such as a determinate and prepositional phrases. A prepositional phrase may begin with one of over fifty prepositions. The learned grammar also includes main verbs and helping verbs. The main verb can be either a linking, action, or a possessive verb, and the helping verbs cover forms of be, forms of do, forms of have, and modal verbs. A noun can be an existing knowledge object within ALPS or one of three pronoun types: demonstrative, personal, and

interrogative pronouns. We have tested our parsing algorithm on various combinations of this learned subset, and for each test sentence, the contents of all the grammar terms have been correctly fulfilled. The rules have been applied and their purposes have been accomplished. Meanwhile, the correct roles are identified and recorded in the parse result.

V. Future work and conclusion

In this paper, we have shown how we have implemented an Object-Oriented program to learn and store a subset of the natural language English. With the use of objects such as grammar terms, structures, roles, and rules, the program can correctly parse sentences belonging to a large variety of sentence structures, and at the same time, identifying the correct purpose for all their components. New grammar terms such as adjectives, adverbs, compound predicates, and gerunds, can easily be taught without the need to change the existing program.

Much Work still needs to be done in order to complete the communication problem. For the second sub-problem of understanding a sentence, the development of the program to execute the recorded roles is currently underway. The third sub-problem to construct a valid sentence reflecting an internal thought is theoretically the reverse process of the second sub-problem. The solutions for both sub-problems are required for the program to communicate. In addition, solutions are needed to handle other grammar usages such as verb tenses, a passive voice, various agreement problems, clauses, and possessive nouns. Furthermore, an important extension is for a communication agent to tolerate sentences with imperfect grammar. Although rules are important in identifying grammatically correct sentences, people do not always speak nor write sentences with perfect grammar. We would like to investigate which rules can be relaxed so that when violated, one can predict the original intention of the writer. Finally, the use of paragraph is required to understand multiple thoughts in a clear and organized manner, and these are the subjects of reading and writing. The successful completion of the communication problem is a first step towards achieving intelligent communication between a computer and a human.

References

[1] S. Russell and P. Norvig, Artificial Intelligence, A Modern Approach, 2nd Ed., Prentice Hall, 2003.

[2] K. Cheng, An Object-Oriented Approach to Machine Learning, Proc. WSES International Conference on Artificial Intelligence, June 2000, pp. 487-492.

[3] K. Cheng, The Representation and Inferences of Hierarchies, Proc. IASTED International Conference on Advances in Computer Science and Technology, January 2006, pp. 269-273.

[4] K. Cheng, Representing Definitions and Its Associated Knowledge in a Learning Program, Proc. International Conference on Artificial Intelligence, June 2007, pp. 71-77.

[5] J. Lambek, The mathematics of sentence structure. Amer. Mathemat. Month. 65, pp. 154-170, 1958.

[6] R. Montague, Universal grammar, Theoria 36, pp. 373-398, 1970.

[7] M. Steedman, Type-raising and directionality in combinatory grammar, Proc. 29th Annual Meeting of the Association for Computational Linguistics, Berkeley, CA, 1991, pp. 71-79.

[8] A. Ranta, Type-Theoretical Grammar, Oxford University Press, Oxford, U.K., 1994.

[9] D.A. Turner, A new implementation technique for applicative languages, Softw. Pract. Exper. 9,1, pp. 31-49, 1979.

[10] P. Hudak, S.L. Peyton-Jones, P. Wadler, B. Boutel, J. Fairbairn, J.H. Fasel, M.M. Guzman, K. Hammond, J. Hughes, T. Johnsson, R.B. Kierburtz, R.S. Nikhil, W. Partain, and J. Peterson, Report on the programming language Haskell, a non-strict, purely functional language, SIGPLAN 27,5, R1-R164, 1992.

[11] R. Leermakers, The Functional Treatment of Parsing, International Series in Engineering and Computer Science, Kluwar Academic Publishers, 1993.

[12] P. Ljunglof, Functional pearls: Functional chart parsing of context-free grammars functional pearl, J. Funct. Program 14,6, pp. 669-680, 2004.

[13] P.C. Callaghan, Generalized LR parsing, The Happy User Guide, Chap. 3, Simon Marlow, 2005.

[14] R. Garigliano, R. Morgan, and M.Smith, The LOLITA system as a contents scanning tool, Proc. 13th International Conference on Artificial Intelligence, Expert Systems and Natural Language Processing, Avignon, France, 1993.

[15] R.A. Frost, W/AGE the Windsor attribute grammar programming environment, Proc. IEEE Symposia on Human Centric Computing Languages and Environments, 2002, pp. 96-99.

[16] R.A. Frost, Realization of Natural Language Interfaces Using Lazy Functional Programming, ACM Computing Surveys, 38,4, Article 11, Dec. 2006.

[17] C. Shan, Monads for natural language semantics, Proc. 13th European Summer School in Logic, Language and Information, Student Session, K. Striegnitz, Ed., Helsinki, Finland, 2001, pp. 285-298.

[18] W. Faris and K. Cheng, A Multi-Dimensional Data Organization that assists in the Understanding and Production of a Sentence, in preparation.