Unknown Oriented Programming: Mathematical Continuation

Zhu Ping*
Tellhow Institute of Smart City, Beijing, China.

* Corresponding author. Tel.: +86 10-59380808; email: zhuping@ tellhow.com.cn
Manuscript submitted March 30, 2021; accepted May 11, 2021.
doi: 10.17706/jsw.16.5.200-207

Abstract: Natural language semantic engineering problems are faced with unknown input and intensive knowledge challenges. In order to adapt to the features of natural language semantic engineering, the AI programing language needs to be expanded mathematically: 1) Using many ways to improve the spatial distribution and coverage of instances; 2) Keeping different abstract function versions running at the same time; 3) Providing a large number of knowledge configuration files and supporting functions to deal with intensive knowledge problems; 4) Using the most possibility priority call to solve the problem of multiple running branches traversal. This paper introduces the unknown oriented programming ideas, basic strategy formulation, language design and simulation running examples. It provides a new method for the incremental research and development of large-scale natural language semantic engineering application. Finally, this paper summarizes the full text and puts forward the further research direction.

Key words: Commonsense, humanoid resolving, semantic engineering, unknown oriented programming

1. Introduction

Machine learning [1] or rule-based reasoning [2] can effectively solve some practical problems when the input scale is limited or the sample features are clear. For example, license plate recognition [3], face recognition [4] or medical diagnosis expert system [5]. However, it is very difficult to deal with the problems such as primary mathematical application problem humanoid resolving [6]-[7], and so on. The input samples are not enough to represent massive language patterns, and it is difficult to cover all execution paths with machine learning or rule reasoning. The existing structured, rule-based or object-oriented programming methods are difficult to deal with the problem with unknown, uncertain, unclear input and inner work flow.

In the case of unknown input range, both samples and logic induction need to solve the problem of input spatial distribution and coverage. For natural language input, the solutions are as follows: 1) The splitting of problem sample clauses and the approximate matching of semantics and patterns, which are the ways to increase the matching scale and degree; 2) The logic abstraction of problem semantic, such as the modification and description of variable meaning and formula commonness, which are the ways to ensure the full space coverage; 3) The logic deduction of samples and resolving patterns, For example, for variable addition and deduction of variable number, this is a combination of sample splitting and logic abstraction. From the perspective of programming language design, these requirements need to add data types, keywords, and even execution statements and common functions to meet the needs of unknown input. It is

even necessary to construct hierarchical data storage mechanism, file type and access interface for the most possibility calculation and reasoning, so as to construct natural language engineering applications.

In the process of research and development, the algorithm model is gradually clear, there is a gradual process, and the implementation program cannot be in place in one step, which is particularly prominent before the mature system is verified. The solution is to keep the running ability of different versions at the same time, and realize the incremental programming processing mechanism. For example, the naming of data element variables can include: the scheme of the nearest noun and quantifier of the data element, the extension of inclusion relationship, or the extension of time concept. Finally, the semantic instance analysis or the comprehensive judgment of the above scheme may be required. In the implementation, the above scheme is not implemented at one time, but is a gradual supplement and improvement. This requires that the program development language has the ability of inheritance and synthesis, and has the thinking analysis method of flexible adjustment function.

Natural language semantic related R&D work is the knowledge intensive project, which need to solve the problem of massive different knowledge representation and activation. The main solutions are as follows: data representation of knowledge, representation of different knowledge as far as possible into standard data configuration files, each file equipped with corresponding activation and derivation program, encapsulated into intelligent program unit by different knowledge and its usage methods. For example, an intelligent program unit composed of mathematical formulas and their applicable conditions. From the perspective of programming language design, these require the development of corresponding data configuration language grammars to represent language semantics and grammar execution semantics; for each data configuration file, the semantic description grammar may be different, and the programming language should provide unified language environment support as far as possible.

In the process of program and algorithm execution, there will be path selection. When selecting, the system will not traverse the execution according to the depth or breadth, but queue up according to the possibility of branches, execute with high possibility, and end the operation when the result is obtained. The possibility first strategy can not only solve the NP cost problem of the traversal space and time, but also solve the problem of knowledge conflict. For example, when matching formula variables and data element variables, the matching degree of tags is the variable matching possibility degree, and the weighted sum of all variables matching possibility degrees is the formula matching possibility degree. The matching possibility degree of the resolving rule determines the running priority. Then the operation space pruning is carried out according to the possibility degree, at the same time, the possibility degree assignment is used to solve the problem of knowledge conflict indirectly, which is a conflict resolution mechanism for unknown oriented programming technology.

2. Basic Strategy

The development of artificial intelligence has proved: Compared with other tools such as computers, human beings are not as powerful as computers in logical reasoning and mathematical calculation. The advantages of human beings lie in memory accumulation, understanding and creative thinking. Generally speaking, human thinking has the characteristics of incremental accumulation, intensive knowledge, comprehensive understanding, shallow logic and calculation, and creativity. In other words, most of the demands of human daily life for thinking are knowledge memory, comprehensive understanding, simple reasoning and calculation, and creative thinking. In order to adapt to the characteristics of human thinking in daily life, the design of humanoid thinking machine should focus on knowledge storage, semantic understanding, simple reasoning and calculation, and the realization mechanism design of creative thinking.

Natural language is the main way for human beings to acquire knowledge. The storage of natural language knowledge can be divided into three types: original input storage, intermediate knowledge storage, reasoning and calculation storage; knowledge understanding is mainly reflected in the correct connection of human like problem processing flow (input, understanding, processing, and output); simple reasoning and calculation are mainly reflected in the implementation of concept matching and simple calculation operation; creative thinking is mainly reflected in the simulation realization, which does the multi-step exploration of case analogy and simple reasoning calculation operation. In this paper, we will take the primary school mathematics application problem as example, focus on the knowledge representation of natural language and recognition, and explore the methods of incremental accumulation, conflict resolution and comprehensive understanding of massive commonsense knowledge.

When the solution of the problem is completely unknown, the most common method is to program and resolve an instance, gradually resolve multiple instances, summarize the law, and abstract the model, so that the final model can resolve all possible inputs of the problem (refer to Fig. 1). In the common programming implementation, the above process is realized by continuously updating the same software system. For natural language semantic engineering problems, it is a project with incalculable cost to abstract a model satisfying all input conditions, let alone iterative optimization. Traditional programming methods are not suitable for natural language semantic engineering problems. It is necessary to explore incremental design to deal with the challenge of unknown oriented programming (UOP).

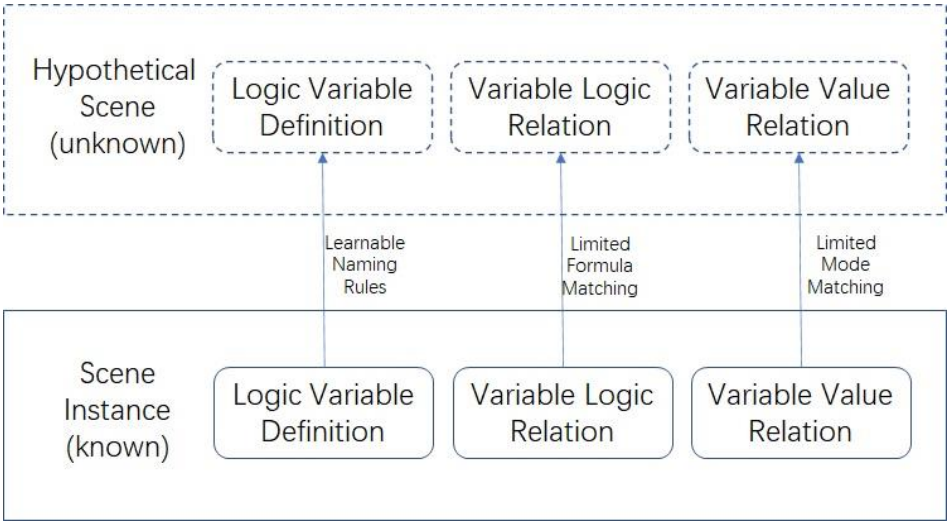

Fig. 1. UOP model of humanoid resolving primary mathematical application problems.

2.1. Strategies for Unknown Input

There are two meaning of unknown input: one is that the input type is changeable and cannot be enumerated and traversed; the other is that due to unpredictable input, the processing flow and function development cannot be achieved in one step, and the processing model is gradually clear with the development process. In view of these two challenges, the strategies of this paper is: the integrated programming language design of case data and processing program, that is, the semantic engineering platform composed of semantic case base, corresponding analogy algorithm, model derivation algorithm, formula library and matching algorithm; The programming idea is to gradually clear the processing model, that is, the evaluation module is initially applicable to specific problems which are matched the possibility degree, the same interface implementation version of algorithm module with different possibility degree is reserved in the running system.

For example, the question: "In the south of the park, a 30 meter long, 0.24 meter wide, and 5 meter high

wall will be built. If 500 bricks are used per cubic meter, how many bricks are needed?"

1) The semantic instance database is established to save the original problem (OP) and ID (ID1), clauses (CL) and their IDs (CID), data element (DE), data element variable (DEV), Location (LC) and Clause number (CN). Tables are designed to store the above information, and the clause matching algorithm, data element identification algorithm, and variable semantic determination algorithm are developed.
2) The model derivation algorithm is established to determine the new features and algorithms of data element identification and variable semantic determination by instances analyzing, including new concept attribute relationship and other forms.
3) build formula library to save the representation of mathematical formula. For example, the formula used in the above instance is as follows: cuboid::length ** cuboid::width ** cuboid::height = cuboid::volume; cuboid::volume ** number::volume::density = total::number. The one-to-one correspondence between data element variable and formula variable is established. For example, the corresponding relationship of variables in the above instance is as follows: wall::long::length, wall::wide::length, wall::high::length, wall::volume --- cuboid::long, cuboid::wide, cuboid::high, cuboid::volume; and wall::volume, brick::number::volume::density, total::brick::number --- cuboid::volume, number::volume::density, total::number. The formula calculating algorithm, variable matching algorithm, and variable correspondence algorithm are established.

The integrated invoking mechanism of modules with different calculation methods for the same function is established. For example, for the data element variable determination algorithm, there are quantifier determination algorithm, time segment division algorithm, concept attribute algorithm, and reference algorithm. These different algorithms constitute the data element determination process, and there may be other more accurate determination algorithms in the follow-up. These algorithms are given different possibilities, which are invoked and run by the central controller according to the maximum possible priority.

2.2. Strategies for Knowledge-Intensive Features

To meet the challenge of knowledge-intensive problem, it is necessary to provide various knowledge configuration files and their matching and identification functions, such as quantifier, concept attribute, time segmentation, reference, additional scene semantics configuration files and access interfaces.

1) Quantifier: Provides the data element type representation information string represented by quantifier, and the matching function between quantifier string and input word.
2) Concept attribute: Represents the concept and its attributes discussed in the problem.
3) Time segmentation: Provides a method to identify data element variables with time feature words.
4) Reference: It means that the concept mentioned previously is briefly mentioned later. Generally speaking, the referential concept inherits the modification semantics of the original concept.
5) Additional scene semantics: Refers to the scene semantics of a problem, which is generally irrelevant to resolve, but helps to understand the problem.

3. Language Design

With the optimization of spatial traversal method as the main clue, the fusion model is designed for the unknown oriented programming, including the integration strategy of gradual clarity (using the strategy of possibility priority or comprehensive judgment of different models), the integration of knowledge-intensive processing (knowledge configuration file integration strategy), the fusion processing of unknown input (semantic circle construction, model meta rule expansion, and formula abstract integration strategy). It forms a programming method characterized by unknown input and workflow, knowledge-intensive and maximum possibility, and takes the possibility priority strategy as the general control mechanism to adapt

to the construction of large-scale complex knowledge reasoning engineering system. Next, this paper discusses in detail five parts: language structure, function call, knowledge configuration, input abstraction, and scene reasoning.

3.1. language Structure

The biggest characteristic of unknown oriented programming language is the expansion of variable structure, which attaches the possibility domain of variable value and three possible alternative values. Assignment statements are divided into: Constant assignment to variable and variable assignment to variable.

3.2. Call Function

The sequence of result transfer between functions, according to the above calculation method of addition possibility assignment function, the possibility value of the parent function is calculated; The parallel relationship of the results between the functions, the different logical clarity degree functions generated in different periods, the functions' possibility values are given different values (limited to the correct programs with different strict conditions), and are linked to the running program chain according to the possibility values, call function with the most possibility value first.

The calling sequence inside the function: nest down the function with the most possibility value in turn. After getting the maximum possibility value of all steps, the value and possibility of this function can be obtained by arithmetic operation. If the result is unsuccessful, the next possibility value of the steps' function will be recalculated until the ideal result is obtained.

3.3. Knowledge Configuration

For the knowledge-intensive feature of programming, many configuration files and their supporting functions are designed. For example, in the function of variable meaning identification, quantifier meaning identification module is implemented first, then concept attribute recognition module and time segment identification module are executed in parallel according to the possibility, and finally optimized by reference relationship identification module.

For unknown-oriented programming languages, it provides a series of toolkits for reading, writing, identifying, and matching the above configuration file, and new configuration file definition method and examples for developing access functions, to facilitate to add and use the new knowledge and commonsense.

3.4. Input Abstraction

For the problem of unknown input, the strategy of this paper is to accumulate examples and classify them one by one according to the problems, clauses, commonsense and numerical explicit transfer relations, split and reorganize, and construct semantic circle; On the basis of semantic circle, explores new model to expand the expression way; With the formulas in primary mathematics knowledge range, extracts clause instances to reorganize, forms a simulation example description of all possible problems. The unknown oriented programming method should include the definition and operation function module of semantic circle, the tool function of extended expression of new model, and the reorganization definition and identification function module of formula description.

3.5. Scene Reasoning

Besides support traditional reasoning and computing functions, unknown-oriented programming cannot do without the support of creative thinking, that is to say, it is necessary to support some thinking jumps. On the one hand, it can be realized by the analogy of commonsense and knowledge to some creative

thinking instances; On the other hand, it needs to support the realization of creative thinking through scene reasoning, such as set and enumeration reasoning. For example, this paper helps to identify a set with the method of identifying the concept attribute relations, and observes and explores the calculation method of The problem by enumerating single instances. For some enumeration examples, the problem can be expanded by means of model expansion. These all need programming language to provide implementation mechanism, supporting tools, and function modules. For example, the identification and judgment of sets, enumeration, observation, reasoning, calculation, and exploration, should be supported with the corresponding common function.

4. Programming Examples

"A commodity box is in the shape of a cube with 6cm edges. How long is the plastic stick needed to make the frame of this box? Put a label around this box, What's the area of the label? "

4.1. Concept Relationship Identification

In the concept relationship identification of data element "what", several functions identify the area attribute. From the process of concept attribute identification of the first two data elements, we can guess that it is the cube surface area. The first feature of data element "how long" concept recognition is "stick", which is identified as "frame" by referring to configuration file. The configuration file of commonsense derivation can be extended to "frame : plastic stick, wooden stick, iron stick", so as to improve the adaptability to unknown input. In the quantifier recognition of data element "6cm", we can infer that the data element is "length" through the quantifier "cm"; We can also infer its attribute as "length" through concept recognition "edge"; At the end of the core word, the first modified concept and the middle attribute recognition, we use a variety of identification methods to comprehensively evaluate the possibility, and obtain the weighted identification possibility of data element concept relationship, which lays the foundation for the possibility priority implementation of the programming controller.

4.2. Dynamic Semantic Circle

Dynamic semantic circle generation needs commonsense support. For example, the cubic power of the cube edge length is equal to the cube volume. It is necessary to build a bridge between the known concept relationship identification results and the resolving formula, so that the corresponding relations between the data elements and the resolving formula variables can be built.

The algorithm compares the matching degree between the antecedent/consequent variable features in the formula configuration file and the variable name composition (core word, first modifier and middle modifier) identified by data element concept relationship. Generally, the simplified method of data element variable name feature including formula variable feature can be adopted.

4.3. Resolving Rule Matching

Resolving rules are predetermined common sense knowledge. The rule base is composed of 6 tuples <ID, EID, MV, RU, CR, PS >. Among them, ID is the identification number of the resolving rule, EID is the identification number of the corresponding formula, RU is the resolving rule, CR is the cancellation action, PS is the possibility of resolving rule. According to the concept relation identification and dynamic semantic circle matching, the activation possibility of corresponding resolving rules is calculated, take the largest possibility rule to execute.

4.4. Resolving Operation

In the three main steps of the problem resolving process, variable concept identification, dynamic

semantic circle generation, and resolving rule matching, the blackboard model can be used to activate the identification function and schedule the operation function by the maximum of possibility.

5. Conclusion

In the case of unknown input range, unknown processing flow, and unknown processing algorithm, humanoid resolving of natural language primary school mathematics application problem should use intensive knowledge to solve simple, limited reasoning steps and computational functions problem. This paper mainly puts forward the method of splitting and reorganizing input examples to solve the mathematical formula expression expansion; proposes the derivation method of commonsense knowledge to solve the default calculation formula problem; proposes the method of pattern analysis expansion to solve the same type of different expression problems and the method of solving mathematical expansion problems. This paper discusses the basic logic calculation construction problem of commonsense knowledge thinking by taking the automatic humanoid resolving of primary mathematical application problems as an example. Firstly, taking the human thinking process as a reference, this paper describes the machine thinking model of commonsense knowledge, designs a variety of commonsense knowledge configuration files and accessing and identifying functions, and explains the automatic humanoid resolving process of primary mathematics application with example. Then with MS VC6.0 language and win10 operating system, the humanoid resolving engine demo is implemented. More than 200 primary mathematics application problems are debugged. The basic logic calculation framework of commonsense knowledge thinking is preliminarily realized, and the effectiveness of the basic principle of commonsense knowledge thinking model is verified. In the next step, this paper will further study the strategy programming, fault tolerant (imprecise) programming, and stage debugging techniques to simulate and adapt to human thinking process.

Conflict of Interest

The author declares no conflict of interest.

References

[1] Ghahramani, Z. (2015). Probabilistic machine learning and artificial intelligence. *Nature, 521(7553)*, 452.

[2] Frye. D., Zelazo, P. D., & Palfai, T. (1995). Theory of mind and rule-based reasoning. *Cognitive Development, 10(4)*, 483-527.

[3] Anagnostopoulos, C. N. E., Anagnostopoulos, I. E., Loumos, V, *et al.* (2006). A license plate-recognition algorithm for intelligent transportation system applications. *IEEE Transactions on Intelligent Transportation Systems, 7*, 377-392.

[4] Zhao, W., Chellappa, R., Phillips, P. J., *et al.* (2003). Face recognition: A literature survey. *ACM Computing Surveys, 35(4)*, 399-458.

[5] Chukwudebe, G. A., Ekwuwune, E., & Nkuma-Udah, K. I. (2018). Medical diagnosis expert system for malaria and related diseases for developing countries. *Proceedings of the IEEE International Conference on Electro-technology for National Development.*

[6] Bilal, S., Amjad, A. S., Omar, A., & Khaled, S. (2017). Arabic arithmetic word problems solver. *3rd International Conference on Arabic Computational Linguistics, Dubai, United Arab Emirates. Proceedings of the Procedia Computer Science, 117* (pp. 153–160).

[7] Sundaram, S. S., & Khemani, D. (2015). Natural language processing for solving simple word problems. *Proceedings of the Twelfth International Conference on Natural Language Processing* (ICON-2015).

Zhu Ping with a doctor's degree, is a senior engineer with the rank of professor, the chief engineer of Tellhow Institute of Smart City. He has been engaged in software development and service more than 20 years. More than 40 scientific and technological papers have been published at home and abroad, and 5 national invention patents have been obtained as the main author. His main areas of interest are thinking machine theory, smart city planning, natural language processing.