Applied and Computational Mathematics
2024, Vol. 13, No. 5, pp. 194-210
https://doi.org/10.11648/j.acm.20241305.18

Research Article

Principle and Application for Rumination Computing Algorithms

Ping Zhu[1, 2, *], Pohua Lv[1], Weiming Zou[3], Xuetao Jiang[1], Jin Shi[3], Yang Zhang[4], Yirong Ma[3]

[1]Beijing Broad Network & Information Company Limited, Beijing, China
[2]Tellhow Institute of Smart City, Beijing, China
[3]Beijing Tellhow Intelligent Engineering Company Limited, Beijing, China
[4]Beijing Yizhuang Smart City Institute Group Company Limited, Beijing, China

Abstract

To fully analyze, mine, and utilize the information and knowledge implied in problem resolving use cases, this paper proposed the autonomous learning method based on machine inducting, hypothesis formulating, and result verifying, which was similar to the biological process of cows ruminating, called rumination computing. Firstly, after inducting and summarizing over 1080 mathematic application problem, the system architecture and general algorithm for humanoid automatic resolving mathematic application problems were represented, which typically included functional modules such as commonsense knowledge base, domain knowledge base, and local knowledge base, preprocessing, word segmentation and part of speech tagging, semantic framework matching, global semantic analyzing, thinking mechanism implementing, etc. Secondly, after the use case solutions were approved, three typical rumination computing modes, including vocabulary sequence, semantic relationship, and computing action, were introduced based on the correct results, resolving steps, and basic rumination actions. The rumination computing step plan was formulated, new knowledge was obtained from the commonsense and results verification, so the continuous autonomous learning loop for machine thinking was formed. Detailed explanations were provided for the three core algorithms implemented (rumination framework algorithm, rumination semantic algorithm, rumination action algorithm). Then, by specific mathematic application problem humanoid resolving user cases, the above three types of rumination computing modes were illumined.

Keywords

Rumination Computing, Mathematic Application Problem Resolving, Explainable Artificial Intelligence, Semantic Understanding, Thinking Mechanism

1. Introduction

The intelligent software systems whose requirement analysis [1-5] executed based on use cases [6-8] were usually carried out through the finite logic extension [9, 10]. This extension could achieve logic coverage of the problem space

*Corresponding author: 1401626437@qq.com (Ping Zhu)

Received: 6 September 2024; **Accepted:** 25 October 2024; **Published:** 29 October 2024

theoretically, and the workflows and results of use cases were usually verified through software debugging. However, all logic branches of the extended part might not be tested and verified [11, 12], and the granularity of the logic extension varied from person to person, place to place, and time to time, without the unified standard and scale. Software quality was closely related to the strength of testing and the ability of developers. In the design and development of large-scale software, intelligent software, and complex logic algorithms, the hidden risks caused by the logic extension became very prominent.

After a use case was solved, the input data "similar to" the use case was input and tested again for "looking back" and "recalling", it was like the human brain to process problems, which was a continuous self-judgment, self-organization, self-improvement, and self-correction for existing results, workflows, information, and knowledge. Just like the process of the cow "ruminating" grasses, first chewing the grasses preliminarily and swallowing them into the stomach for initial digestion. In the idle time, the partially digested food in the stomach could be turned back into the mouth for fine chewing again. In task-oriented software design, the function development should be completely in the fastest and most cost-effective way; During system testing, the input data and constraint logic of the test cases were extended. With the support of software self-sensing mechanisms such as trend prediction and commonsense judgment of step results, the context logic boundaries of the derived conclusions were examined. This machine learning technique [13-15] was used to evaluate the applicability of the system processing methods and explore new resolving methods, which were called "rumination computing".

Rumination computing played a particularly significant role in resolving mathematic application problems and was the important study experience of the excellent students. The precise analysis of the applicability conditions for the formulas and resolving methods, the derivation and verification for new resolving paths, and even the proposal and verification of new theorem, had great significance for the machine learning. Now the big data [16, 17] became the mainstream technology for research and development, based on machine rumination computing technology, it had a very broad application prospect to achieve logic coverage of problem space by use case big data; The use cases used for model training [18] could also be filtered by rumination computing techniques to reduce the logically duplicated samples while ensuring logic coverage; Use cases with the same resolving features could be limited in extension and induction of the input data element attributes, forming higher-level feature hypotheses, and using use cases that satisfied the feature hypotheses to verify the resolving method, thereby expanding the problem space coverage of the use cases; Based on the correct results of use cases, it was possible to discover and validate new resolving paths by combining the set of possible resolving actions and the original input data elements for iterative traversal. This article would explore the use of machine rumination computing methods in the field of automatic humanoid resolving for mathematic application problems [19-22]. Through results verification [23, 24], commonsense judgment [25], and trend prediction [26], and other system self-perception mechanisms [27], hypotheses might be automatically proposed and self-verified, constructing a machine autonomous intelligent system self-improvement closed loop with use case big data [28-30].

2. General Humanoid Resolving System

The general humanoid automatic resolving system for mathematic application problems [31] typically included function modules such as commonsense knowledge base, domain knowledge base, and local knowledge base [32], preprocessing, word segmentation and part of speech tagging [33], semantic framework matching [34, 35], global semantic analyzing [36], thinking mechanism implementing, etc. (refer to figure 1).

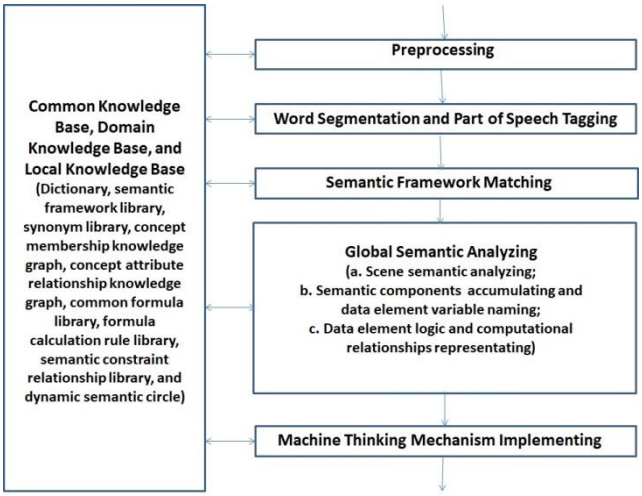

Figure 1. *System architecture for general humanoid automatic resolving mathematic application problem.*

2.1. Knowledge Base

The semantics of corpus resources were tagged formally to store in knowledge base, in addition, the commonsense knowledge base, domain knowledge base, and local knowledge base were used to store background data, information, and knowledge for use case computing and deduction. The commonsense knowledge base included thinking machine dictionary, semantic framework base, synonym base, concept membership knowledge graph, concept attribute relationship knowledge graph, common formula library, formula computation rule library, semantic constraint relationship base, etc. The domain knowledge base contained domain knowledge and was used to overload commonsense

knowledge. Local knowledge base referred to temporary and scene data, information, and knowledge related to problem resolving.

2.2. General Resolving Algorithm

Input: Problem text *Text*, dictionary *Dict*, semantic framework base *SFBase*, commonsense knowledge base *CKBase*, domain knowledge base *FKBase*, etc;

Output: Preprocessing result *Text'*, step results for problem resolving *ANOutputList*, step results for word segmentation and part of speech *WRList*, matched semantic frameworks for problem *SFList*, semantic components *SCTable*, scene identifiers sequence *SIDList*, data element variables *DETable*, dynamic semantic circle *DSCircle*, etc.

Algorithm: General_Algorithm (*Text*, *Dict*, *SFBase*, *CKBase*, *FKBase*)

1) *Text'*<--- preprocess *(Text, Dict);*

///Problem text preprocessing included symbol system standardization, unit base unification, named entity recognition, etc.

2) WRList<--- word-segmentation&part-of-speech-tagging *(Text', Dict);*

///Word segmentation and part of speech tagging functions were performed through modules such as dictionary retrieval, vocabulary sequence correction, vocabulary pattern correction, part of speech sequence correction, and part of speech pattern correction, as well as corresponding data, and commonsense knowledge.

3) *SFList*<---semantic_framework_matching *(WRList, SFBase);*

/// After automatic word segmentation and part of speech tagging, semantic framework matching was used to recognize pattern WRList of input clause with semantic frameworks in SFBase. The multidimensional scene semantics of the preceding matched semantic framework were inherited or overloaded to implement semantic information supplementation across the semantic frameworks, and knowledge such as synonym, equivalent representation structure base, and semantic framework description grammar were used to determine the multidimensional formal semantics contained in the text, forming a complete formal semantic representation of the clause. In terms of automatic humanoid resolving of mathematic application problems, the formal semantic representation included multidimensional semantics of skeleton vocabularies sequence, such as general formulas, logic condition constants, logic condition formulas, and logic inequality relationships.

4) *SIDList*<--- global_semantic_analysisA *(SFList, CKBase, FKBase);*

/// The stage A of global semantic analysis focused on the scenes. To distinguish and identify different semantic scenes, the stage A constructed the scene feature word strings in clause by words that could be inherited, overloaded, or ignored.

5) SCTable<--- global_semantic_analysisB *(WRList, SFBase, CKBase, FKBase);*

DETable<--- global_semantic_analysisB *(WRList, SFBase, CKBase, FKBase);*

///The stage B of global semantic analysis focused on the semantic component accumulating and the data element variable naming. The overall semantic formal representation of the problem might be accumulated and condensed by semantic components; The data element variable naming assigned name by global semantic. Firstly, it was necessary to determine the parallel relationship between clause scene identifiers. If all clause scene identifiers were parallel, semantic overloading technology should be used to name the data element variables. If there were subordinate relationship scenes contained in the upper and lower scene identifiers, semantic inheritance technology should be used to form a global scene identifier vocabulary string, and it should be compared with the local semantic variable name in sequence. If there were words that should be overloaded (parallel relationship words with common upper concepts in the knowledge graph), the parallel relationship words should be replaced according to the overloading technology; Otherwise, should inherit the global scene identification word and form the global semantic names of the data element variables (all identifiers were arranged in a string from the large concept extension to small one in sequence).

6) DSCircle<--- global_semantic_analysisC *(WRList, DETable, SFBase, CKBase, FKBase);*

///The stage C of global semantic analysis focused on explicit representation of calculation and logic relationships. The explicit representations of calculation and logic relationships were used to represent the explicit/implicit calculation and logic relationships between data element variables contained in the context of natural language. The types of data element variables included not only explicit known data element variables and question data variables, but also implicit intermediate data element variables. There were also two ways of representing calculation and logic relationships, explicit and implicit, both of which were represented in a standard form, corresponding to calculation formulas and their resolving rules, forming dynamic data, information, and knowledge pool (dynamic semantic circle) that might be used for problem resolving, in order to enable machine thinking mechanisms to achieve derivation and calculation results.

7) *ANOutputList*<--- thinking_mechanisms *(DETable, DSCircle);*

///This module invoked the thinking mechanism to resolve and output the derivation and calculation results of each step. The machine thinking mechanisms were used to iteratively and comprehensively computing and deduce based on data element variables, calculation rules, and corresponding resolving formulas, by different thinking mechanisms, until all problem variables were resolved. For example, equation resolving could be used to sequentially set the question data

element variable as x, and then used calculation rules and corresponding resolving formulas to inference until all question variables were resolved by the equation; It was also possible to iteratively enumerate the values of data element variables under feature constraints based on the feature patterns in the semantic constraint base, and explore solutions until the answer to the question variable satisfied the constraint conditions.

3. Rumination Computing Implementation

Most machine learning algorithms fitted the logic implied in static datasets, such as the maximum entropy algorithm [37]. Inspired by the ChatGPT [38] autoregressive prediction algorithm, this paper proposed the commonsense verification logic loop for rumination computing, which meant that after the resolving scheme was confirmed, the middle steps such as the resolving steps and the applicable conditions of formulas were evaluated based on the standard answer again, and the data-driven rumination computing step plan was formulated. From beginning to end, new knowledge might be continuously obtained from the affirmation of commonsense and verification, forming the continuous autonomous closed loop machine thinking mechanism.

3.1. Rumination Computing Workflows

The problem solved with the correct result was the premise for rumination computing, so the correct result should be the important criterion for judging the success of rumination computing. Rumination computing was the functional extension for basic machine thinking mechanisms, forming the composite self-learning machine thinking mechanism suitable for complex scenarios. Rumination computing could determine the applicable conditions for general formula, the extension scale for the variable retrieve pattern, and verifying hypotheses for new resolving ideas. The rumination computing workflow had three branches (rumination modes) A, B, and C (refer to figure 2):

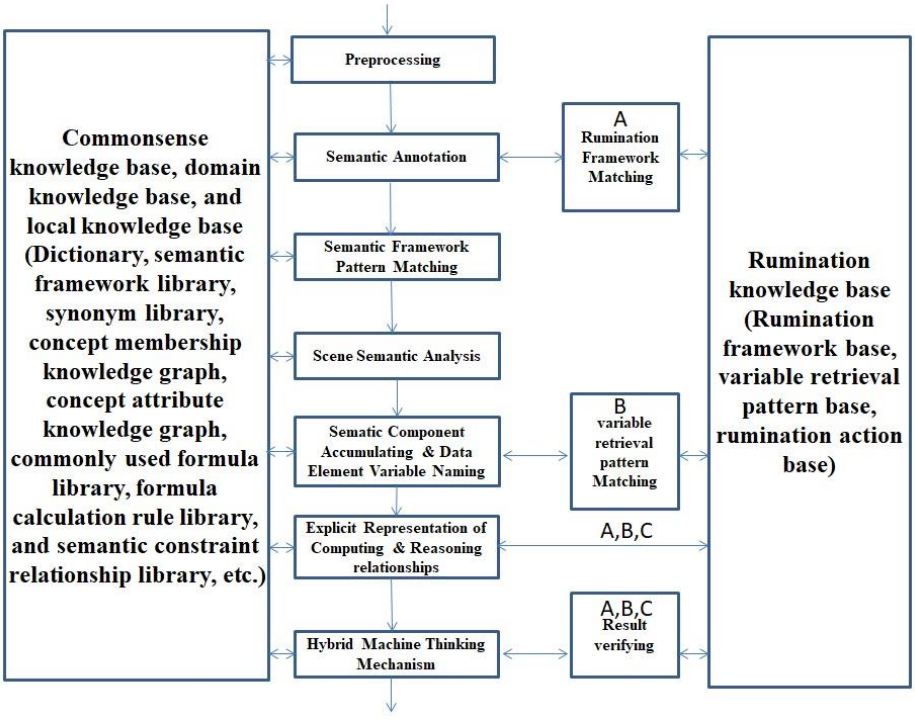

Figure 2. *Hybrid-modes machine thinking.*

3.1.1. Rumination Mode A

Extracted and generated rumination frameworks (feature vocabulary and attribute sequences) from problem text. Based on synonym, concept attribute relationship graph, and other commonsense knowledge, the vocabulary and attribute sequence obtained by the segmentation and part of speech tagging was matched with the rumination frameworks. the pre-set algorithm corresponding to the rumination framework was applied to generate rumination action plans, and the action plans were executed separately. Output the step data and result data recorded by the system, and analyzed and judged the output data based on commonsense knowledge and semantic constraints of the action plans. For example, commonsense such as "there are no negative numbers in the semantic expression of the objective world" and "the divisor

cannot be zero" could be used to determine the "affirmative" error of the result; The constraint on the interval values of data element variables could determine whether the data element variables complied with the constraint, or their values were out of bounds.

3.1.2. Rumination Mode B

When the problem was solved correctly, the problem' semantic components in the context, as well as known data element variables, middle data element variables, calculation and logic relationship sets, etc., were extracted to form the semantic component constraint (replaceable and non-replaceable) framework that could be applied and extended, namely the variable retrieval pattern. For new input problems, under the condition of satisfying the variable retrieval pattern (i.e. satisfying the semantic components and known data element variable constraints), middle data element variables and their related calculation and logic relationship hypotheses were generated, and the correctness of the resolving method hypotheses could be verified through the problem result.

3.1.3. Rumination Mode C

Based on the rumination action set (included the basic deduction and calculation actions such as addition, subtraction, multiplication, division, rollback, etc.), the resolving engine should achieve the same correct result with the general problem resolving system when the other correct basic action sequences were explored.

3.2. Core Algorithms

3.2.1. Rumination Framework Algorithm (Mode A)

Input: problem text *Text*, dictionary *Dict*, semantic framework base *SFBase*, commonsense knowledge base *CKBase*, domain knowledge base FKBase, rumination knowledge base RMBase, etc.

Output: Problem solving answer *ANOutputList*, word segmentation and part of speech *WRList*, mode A rumination framework *RMA_A_TYPE* for problem matching, rumination action planning *RMActionPlan*, data element variable table *DETable*, dynamic semantic circle DSCircle, etc.

Algorithm description: Rumination_Algorithm_A (*Text, Dict, SFBase, CKBase, FKBase, RMBase*)

1) *ANOutputList, WRList, RM_A_TYPE, DSCircle* <--- General_Algorithm *(Text, Dict, SFBase, CKBase, FKBase);*
   /// Using the general algorithm in section 2.2, resolved the problem and verified the result. If the result was correct, output *ANOutputList, WRList, RM_A_TYPE,* and *DSCircle;* If the result was incorrect, exited the rumination framework algorithm.
2) *RMActionPlan*<--- pattern_match*(WRList, RM_A_TYPE, RMBase);*
   /// If the problem matched the mode A rumination framework *RM_A_TYPE* (constructed by the vocabulary and attribute sequence), output the rumination action plan; If the problem did not match, exited the rumination algorithm.
3) *DETable*<--- asign_data_element_vars_values *(DETable, RMActionPlan, RMBase);*
   /// According to the rumination action plan, for the relevant data element variables, reassigned values which satisfied with the feature constraints.
4) *ANOutputList*<--- thinking_mechanisms*(DETable, DSCircle);*
   ///Re-invoked the thinking mechanism to resolve the problem and output each step's results.
5) *If(*is_satisfy_RMStrict*(RMActionPlan)) goto* ③ *;*
   ///According to the rumination action plan, jumped to step ③, for the relevant data element variables, iteratively assigned values, and accumulated the result data of each resolving step.
6) verify_answers*(ANOutputList, CKBase, FKBase, RMBase);*
   ///After the rumination action plan was executed completely, the results were cross validated. If there were conflicts or errors in commonsense or domain knowledge, the applicable conditions of the relevant computing actions were analyzed, recorded, and stored.

3.2.2. Rumination Semantic Algorithm (Mode B)

Input: problem text *Text and Text_New*, dictionary *Dict*, semantic framework base *SFBase*, commonsense knowledge base *CKBase*, domain knowledge base FKBase, rumination knowledge base RMBase, etc.

Output: Problem solving answer *ANOutputList*, word segmentation and part of speech *WRList*, semantic framework *SFList*, type B rumination component framework *RM_B_TYPE and RM_B_TYPE'*, rumination action planning *RMActionPlan*, data element variable tables *DETable* and *DETable',* dynamic semantic circles *DSCircle* and *DSCircle'*, etc.

Algorithm description: Rumination_Algorithm_B (*Text, Dict, SFBase, CKBase, FKBase, RMBase*)

1) *ANOutputList* <--- General_Algorithm *(Text, Dict, SFBase, CKBase, FKBase);*
   IF (! verified_right*(ANOutputList)) RETURN FALSE;*
   ELSE RM_B_TYPE<--- record_resolving_method *(DETable, DSCircle, SCFrame);*
   /// Using the general algorithm in section 2.2, resolved the problem and verified the result. If the result was correct, output *ANOutputList* and *RM_B_TYPE;* If the result was incorrect, exited the rumination semantic algorithm.
2) *WRList*<---segmentation&part-of-speech *(Text_New);*
   ///Performed word segmentation and part of speech tagging on the new problem *Text_New*;
3) *SFList*<---semantic_framework_match*(WRList,*

SFBase);
/// Generated the semantic framework sequences *SFList* for the new problem;

4) *RM_B_TYPE'* <---global_semantic_analysis*(SFList, CKBase, FKBase);*
/// Generated the semantic component framework for the new problem;

5) *RMActionPlan*<--- pattern_match *(RM_B_TYPE, RM_B_TYPE', RMBase);*
/// If the new problem *RM_B_TYPE'* matched the mode B rumination framework *RM_B_TYPE* (constructed by the semantic components of the result verified problem), output the rumination action plan; If the problem did not match, exited the rumination semantic algorithm.

6) *DETable'* <--- repalce_semantic_components *(DETable, RMActionPlan, RMBase);*
/// According to the rumination action plan of the mode B, replaced the semantic components of the relevant variables in the data element variable table of new problem.

7) *DSCircle'* <---repalce_vars_names *(DETable, DETable', DSCircle);*
/// Replaced the variable names in the old dynamic semantic circle based on the correspondence between variables in the old and new data element variable tables.

8) *ANOutputList'* <---thinking_mechanisms *(DETable', DSCircle');*
/// Invoked the thinking mechanism to resolve new problems and output the results of each step.

9) *ANOutputList''* <--- General_Algorithm *(Text_New, Dict, SFBase, CKBase, FKBase);*
/// Solved the new problem using general resolving algorithms and output the results of each step.

10) verify_answer*(ANOutputList', ANOutputList'', CKBase, FKBase, RMBase);*
/// Verified the results. If there were no conflicts or errors in commonsense or domain knowledge, and the results were consistent, recorded the relevant rumination semantic component framework *RM_B_TYPE* as knowledge. Otherwise, removed the *RM_B_TYPE* from the *RMBase*.

3.2.3. Rumination Action Algorithm (Mode C)

Input: problem text *Text*, dictionary *Dict*, semantic framework base *SFBase*, commonsense knowledge base *CKBase*, domain knowledge base FKBase, rumination knowledge base *RMBase*, rumination action set *RASet*, etc.

Output: Problem solving answer *ANOutputList*, word segmentation and part of speech *WRList*, data element variable table *DETable*, rumination action step results *RAResult*, the rumination action sequence *RAList* of the correct result, etc.

Algorithm description: Rumination_Algorithm_C *(Text, SFBase, CKBase, FKBase, RMBase, RASet)*

1) ANOutputList<--- General_Algorithm *(Text, Dict, SFBase, CKBase, FKBase);*
/// Using the general algorithm in section 2.2, resolved the problem and verified the result. If the result was correct, output *ANOutputList;* If the result was incorrect, exited the rumination action algorithm.

2) *RAResult*<---iteral_explore*(DETable, RASet);*
///Iteratively selected the executable computing action, and stored the computing results in *RAResult*;

*3) IF(*is_verified_results*(ANOutputList, RAresult)) THEN RALists*<--- output_action_sequence*(DETable, RASet);*
///If the rumination computing result was consistent with the general algorithm result, output the rumination computing action path;

4) *ELSE goto ②;*
///Otherwise, iteratively explored other variables and computing actions.

4. Data Analysis

4.1. Word Sequence Instances

4.1.1. Instance A

"Between one o'clock and two o'clock (C1), when do minute hand and hour hand form a right angle (C2) ?"

1) Word sequence rumination framework

RMA_A_TYPE 4_1_1 /// Rumination framework identifier (ID)

time interval /// Rumination pattern type

Starting~time~::Hour~hand~, Ending~time~::Hour~hand~, ///Rumination variables

one o'clock, two o'clock, /// Original values of rumination variables

Starting~time~::Hour~hand~, zero o'clock, twelve o'clock, /// Rumination variable names and possible ranges of values

Ending~time~::Hour~hand~, zero o'clock, twelve o'clock, /// Rumination variable names and possible ranges of values

Time::Interval, one o'clock, /// Rumination time interval value

equal_to, more_than, less_than, /// Rumination logic relationship

integer, positive~number~, negative~number~, real~number~, odd~number, even~number~ /// Rumination commonsense attributes

2) Rumination action plan

According to the semantic frameworks (vocabulary sequence) and the rumination framework *RMA_A_TYPE4_1_1*, the following rumination computing action plan could be generated, where the rumination variable "*Starting~time~::Hour~hand~*" was assigned values from 0 to 11 with the interval of 1 hour; And the "*Ending~time~::Hour~hand~*" and "*Starting~time~::Hour~hand~*" were assigned values with the interval of 1 hour, then the system was resolved by the general

(original) algorithm.

3) Semantic annotation

After clause segmentation, word segmentation, and part of speech recognition, the input clauses were matched with the following semantic frameworks to generate the problem's data element variable table and dynamic semantic circle.

C1 ///semantic framework ID

General /// semantic framework type

6 /// semantic framework item quantity

t Between /// "t" represented other word types.

n Starting~Time~::Hour~Number~ /// "n" represented data element variable; "Starting~Time~::Hour~Number~" was the local semantic name;

q o'clock /// "q" represented quantities.

t and

n Ending~Time~::Hour~Number~

q o'clock ///

C2 /// semantic framework ID

Expanding ///semantic framework types

10/// general formula quantity

A::Right~Angle~::Minute~Number~,A::Right~Angle~::Minute~Number~,60,///Data element variables

3 ///formula ID

1_Vaviable,2_Vaviable,Times, ///formula variables

B::Right~Angle~::Minute~Number~,B::Right~Angle~::Hour~Number~,60,/// Data element variables

3 ///formula ID

1_Vaviable,2_Vaviable,Times, ///formula variables

A::Result::Hour~Number~,A::Right~Angle~::Hour~Number~,Starting~Time~::Hour~Number~,/// Data element variables

6 ///formula ID

Total,PartA,PartB, /// formula variables

B::Result::Hour~Number~,B::Right~Angle~::Hour~Number~,Starting~Time~::Hour~Number~,/// Data element variables

6 ///formula ID

Total,PartA,PartB, /// formula variables

6,Minute~Hand~::Tracing::Speed,0.5,/// Data element variables

6 ///formula ID

Total,PartA,PartB, /// formula variables

A::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~,A::Right~Angle~::Minute~Number~,Minute~Hand~::Tracing::Speed, /// Data element variables

3 ///formula ID

1_Vaviable,2_Vaviable,Times, ///formula variables

B::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~,B::Right~Angle~::Minute~Number~,Minute~Hand~::Tracing::Speed, /// Data element variables

3 ///formula ID

1_Vaviable,2_Vaviable,Times, ///formula variables

Hour~Hand~::Starting~Time~::Degree~Number~,Offset::Degree~Number~,Minute~Hand~::Starting~Time~::Degree~Number~, /// Data element variables

6 ///formula ID

Total,PartA,PartB, /// formula variables

A::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~,Offset::Degree~Number~,90, /// Data element variables

6 ///formula ID

Total,PartA,PartB, /// formula variables

270,B::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~,Offset::Degree~Number~,/// Data element variables

6 ///formula ID

Total,PartA,PartB, /// formula variables

2 /// The condition formula quantity of the framework

Existed,Starting~Time~::Hour~Number~,;Un-Existed, Starting~Time~::Minute~Number~,;/// the preconditions of formula

Hour~Hand~::Starting~Time~::Degree~Number~,Starting~Time~::Hour~Number~,30, ///Data element variables

3 ///formula ID

1_Vaviable,2_Vaviable,Times, ///formula variables

Existed,Starting~Time~::Hour~Number~,;Un-Existed, Starting~Time~::Minute~Number~,; /// the preconditions of formula

0,Starting~Time~::Minute~Number~, /// data element variables

9 /// formula ID

Left,Right, /// formula variables

0 ///The logic relationships quantity of the semantic framework

20/// Semantic framework item quantity

n A::Result::Hour~Number~ /// "n" represented data element variable; "A::Result::Hour~Number~" represented the local semantic name.

w B::Result::Hour~Number~ /// "w" represented question data element variable; "A::Result::Hour~Number~" represented the local semantic name.

v do /// "v" represented verb.

s minute hand///"s" represented noun.

t and /// "t" represented other types of words.

s hour hand

v form

c a /// "a" represented constant.

s right angle

m A::Right~Angle~::Minute~Number~ /// "m" represented middle data element variable.

m A::Right~Angle~::Hour~Number~

m B::Right~Angle~::Minute~Number~

m B::Right~Angle~::Hour~Number~

m Minute~Hand~::Tracing::Speed

m A::Minute~Hand~::Right~Angle~::Tracing:: Degree~Number~

m B::Minute~Hand~::Right~Angle~::Tracing:: Degree~Number~

m Hour~Hand~::Starting~Time~::Degree~ Number~

m Minute~Hand~::Starting~Time~::Degree~ Number~

m Offset:: Degree~Number~
m Starting~Time~::Hour~Number~

4) Data element variable table

After word segmentation and part of speech tagging of the input text, pattern matching was performed with the semantic frameworks. Through global semantic analysis, the following data element variable table could be generated (refer to Table 1):

Table 1. *Data element variable table.*

No	Type	Name	Value
1	Constant	Starting~Time~::Hour~Number~	1.000000
2	Constant	Ending~Time~::Hour~Number~	2.000000
3	Question Variable	A::Result::Hour~Number~	Unknown
4	Question Variable	B::Result::Hour~Number~	Unknown
5	Middle Variable	A::Right~Angle~::Minute~Number~	Unknown
6	Middle Variable	A::Right~Angle~::Hour~Number~	Unknown
7	Middle Variable	B::Right~Angle~::Minute~Number~	Unknown
8	Middle Variable	B::Right~Angle~::Hour~Number~	Unknown
9	Middle Variable	Minute~Hand~::Tracing::Speed	Unknown
10	Middle Variable	A::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~	Unknown
11	Middle Variable	B::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~	Unknown
12	Middle Variable	Hour~Hand~::Starting~Time~::Degree~Number~	Unknown
13	Middle Variable	Minute~Hand~::Starting~Time~::Degree~Number~	Unknown
14	Middle Variable	Offset::Degree~Number~	Unknown
15	Constraint Variable	Starting~Time~::Hour~Number~	Minimum
16	Constraint Variable	Ending~Time~::Hour~Number~	Maximum

Notes: The fifteenth and sixteenth variables were constraint variables of the same name, with their respective minimum and maximum values.

5) Dynamic semantic circle

The formula antecedents in the dynamic semantic circle were shown in following Table 2:

Table 2. *Formula antecedent table.*

ID	Antecedent
11	Existed, Starting~Time~::Hour~Number~,; UnExisted, Starting~Time~::Minute~Number~,;
12	Existed, Starting~Time~::Hour~Number~,; UnExisted, Starting~Time~::Minute~Number~,;

The formulas in dynamic semantic circle were shown in following Table 3:

Table 3. *Formula table.*

ID	Formula ID	Data Element Variable	Formula Variable
1	3	A::Right~Angle~::Minute~Number~, A::Right~Angle~::Hour~Number~,60,	1_Vaviable,2_Vaviable, Times,
2	3	B::Right~Angle~::Minute~Number~, B::Right~Angle~::Hour~Number~,60,	1_Vaviable,2_Vaviable, Times,
3	6	A::Result::Hour~Number~, A::Right~Angle~::Hour~Number~, Offset::Hour~Number~,	Total, PartA, PartB,
4	6	B::Result::Hour~Number~, B::Right~Angle~::Hour~Number~, Offset::Hour~Number~,	Total, PartA, PartB,
5	6	6, Minute~Hand~::Tracing::Speed, 0.5,	Total, PartA, PartB,
6	3	A::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~, A::Right~Angle~::Minute~Number~, Minute~Hand~::Tracing::Speed,	1_Vaviable, 2_Vaviable, Times,
7	3	B::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~, B::Right~Angle~::Minute~Number~, Minute~Hand~::Tracing::Speed,	1_Vaviable, 2_Vaviable, Times,
8	6	Hour~Hand~::Starting~Time~::Degree~Number~, Offset::Degree~Number~, Minute~Hand~::Starting_Time::Degree~Number~,	Total, PartA, PartB,
9	6	A::Minute~Hand~::Right~Hand~::Tracing::Degree~Number~, Offset::Degree~Number~,90,	Total, PartA, PartB,
10	6	270, B::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~, Offset::Degree~Number~,	Total, PartA, PartB,
11	3	Hour~Hand~::Starting~Time~::Degree~Number~, Starting~Time~::Hour~Number~,30,	1_Vaviable, 2_Vaviable, Times,
12	9	0, Minute~Hand~::Starting~Time~::Degree~Number~,	Left, Right,

6) Resolving and analyzing

Based on the formulas listed above, the rumination action plan was formulated for instances that satisfied the rumination framework of the feature words sequence. The rumination action plan was used to resolve the problem iteratively (refer to Table 4).

Table 4. *Instance A step results.*

Step	Variable	*0-1o'clock*	*1-2o'clock*	*2-3o'clock*	*3-4o'clock*	4-5*o'clock*	5-6 *o'clock*	6-7 *o'clock*	7-8 *o'clock*	8-9 *o'clock*	9-10 *o'clock*	10-11 *o'clock*	11-12 *o'clock*
1	Minute~Hand~::Starting~Time~::Degree~Number~	*0*	*0*	*0*	*0*	0	0	0	0	0	0	0	0
2	Hour~Hand~::Starting~Time~::Degree~Number~	*0*	*30*	*60*	*90*	120	150	180	210	240	270	300	330
3	Minute~Hand~::Tracing::Speed	*5.5*	*5.5*	*5.5*	*5.5*	5.5	5.5	5.5	5.5	5.5	5.5	5.5	5.5
4	Offset::Degree~Number~	*0*	*30*	*60*	*90*	120	150	180	210	240	270	300	330
5	A::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~	*90*	*120*	*150*	*180*	210	240	270	300	330	360	390	420
6	B::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~	*270*	*240*	*210*	*180*	150	120	90	60	30	0	-30	-60
7	A::Right~Angle~::Minute~Number~	*16.36*	*21.82*	*27.27*	*0*	38.18	43.64	49.09	54.55	60	65.45	70.91	76.36
8	B::Right~Angle~::Minute~Number~	*49.09*	*43.64*	*38.18*	*32.73*	27.27	21.82	16.36	10.91	5.45	0	-5.45	-10.91

Step	Variable	*0-1o'clock*	*1-2o'clock*	*2-3o'clock*	*3-4o'clock*	4-5*o'clock*	5-6 *o'clock*	6-7 *o'clock*	7-8 *o'clock*	8-9 *o'clock*	9-10 *o'clock*	10-11 *o'clock*	11-12 *o'clock*
9	A::Right~Angle~::Hour~Number~	*0.27*	*0.36*	*0.45*	*32.73*	0.64	0.73	0.82	0.91	1	1.09	1.18	1.27
10	B::Right~Angle~::Hour~Number~	*0.82*	*0.73*	*0.64*	*0.55*	0.45	0.36	0.27	0.18	0.09	0	-0.09	-0.18
11	A::Result::Hour~Number~	*0.27*	*1.36*	*2.45*	*3.55*	4.63	5.72	6.82	7.91	9	10.09	11.18	12.27
12	B::Result::Hour~Number~	*0.82*	*1.73*	*2.64*	*3.55*	4.45	5.36	6.27	7.18	8.09	9	9.91	10.82
Results number		*2*	*2*	*2*	*1*	2	2	2	2	2	1	0	0

The rumination computing results of Instance A should satisfy the default relationship: " A::Result::Hour~Number~" was less than "B::Result::Hour~Number~"; The following conditions should also be met: The results should be within the time interval of the rumination action plan; It must also meet commonsense: The variables should be positive numbers or zero, and all degrees should be between 0-360 degrees (determined by the constraint condition time interval). Therefore, the correct result range of Instance A could be preliminarily identified as the first four input scenarios (italicized data columns), where the starting times were between the intervals [0,3]. The specific conclusion could be obtained through cross validating results of other instances.

4.1.2. Instance B

“Between five o'clock and six o'clock (C1), when do minute hand and hour hand form a right angle (C2)?”

The feature word sequence of the semantic frameworks for matching the instance B was the same as instance A, however, the input data element variable values were different, and the preset formulas for the dynamic semantic circle were also different. Table 5 showed the information of data element variables that were different from instance A (the same information was omitted), while Table 6 showed the formula information in the dynamic semantic circle that was different from instance A (the same information was omitted).

Table 5. *Data element variables.*

NO	Type	Name	Value
1	Constant	Starting~Time~::Hour~Number~	5
2	Constant	Ending~Time~::Hour~Number~	6
…	…	…	…

Table 6. *Formula table.*

ID	Formula ID	Data Element Variable	Formula Variable
…	…	…	…
9	6	Offset::Degree~Number~, A::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~,90,	Total, PartA, PartB,
10	6	B::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~,90,	Total, PartA, PartB,
…	…	…	…

The step results were shown in Table 7:

Table 7. *Instance B step results.*

Step	Vaviable	*0-1 o'clock*	*1-2 o'clock*	*2-3 o'clock*	*3-4 o'clock*	*4-5 o'clock*	*5-6 o'clock*	*6-7 o'clock*	*7-8 o'clock*	*8-9 o'clock*	*9-10 o'clock*	*10-11 o'clock*	*11-12 o'clock*
1	Minute~Hand~::Starting~Time~::Degree~Number~	0	0	0	0	*0*	*0*	*0*	*0*	0	0	0	0
2	Hour~Hand~::Starting~Time~::Degree~Number~	0	30	60	90	*120*	*150*	*180*	*210*	240	270	300	330
3	Minute~Hand~::Tracing::Speed	5.5	5.5	5.5	5.5	*5.5*	*5.5*	*5.5*	*5.5*	5.5	5.5	5.5	5.5
4	Offset::Degree~Number~	0	30	60	90	*120*	*150*	*180*	*210*	240	270	300	330
5	A::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~	90	120	150	180	*210*	*240*	*270*	*300*	330	360	390	420
6	B::Minute~Hand~::Right~Angle~::Tracing::Degree~Number~	270	240	210	180	*150*	*120*	*90*	*60*	30	0	-30	-60
7	A::Right~Angle~::Minute~Number~	16.36	21.82	27.27	0	*38.18*	*43.64*	*49.09*	*54.55*	60	65.45	70.91	76.36
8	B::Right~Angle~::Minute~Number~	49.09	43.64	38.18	32.73	*27.27*	*21.82*	*16.36*	*10.91*	5.45	0	-5.45	-10.91
9	A::Right~Angle~::Hour~Number~	0.27	0.36	0.45	32.73	*0.64*	*0.73*	*0.82*	*0.91*	1	1.09	1.18	1.27
10	B::Right~Angle~::Hour~Number~	0.82	0.73	0.64	0.55	*0.45*	*0.36*	*0.27*	*0.18*	0.09	0	-0.09	-0.18
11	A::Result::Hour~Number~	0.27	1.36	2.45	3.55	*4.63*	*5.72*	*6.82*	*7.91*	9	10.09	11.18	12.27
12	B::Result::Hour~Number~	0.82	1.73	2.64	3.55	*4.45*	*5.36*	*6.27*	*7.18*	8.09	9	9.91	10.82
Results number		2	2	2	1	*2*	*2*	*2*	*2*	2	1	0	0

The rumination computing results of Instance B should satisfy the default relationship: " A::Result::Hour~Number~" was less than "B::Result::Hour~Number~"; The following conditions should also be met: The results should be within the time interval of the rumination action plan; It must also meet commonsense: The variables should be positive numbers or zero, and all degrees should be between 0-360 degrees (determined by the constraint condition time interval). Therefore, the correct result range of Instance B could be preliminarily identified as the middle four input scenarios (italicized data columns), where the starting times were between the intervals [0,3]. The specific conclusion could be obtained through cross validating results of other instances.

4.2. Semantic Relation Instance

The semantic components of text (subject, object, action and attributes, time and place, etc.) and their relationships, as well as the more, less, or proportion relations of data elements, together with system data, constituted another type of rumination framework summarized from sample data. After verification, it could form the new thinking mechanism of feature analogy.

For example, two subject EN "A:: B::", the action AC "Run", the place PL "Ring::Runway::", the time AT "First::Meet::Time::", the action modifier string AB "Simultaneous~Departure~::Same~Starting~Place~::Same~Direction::", the subject attribute string ER "A(person)::B (person)::", the place modifier string AP "Ring::", and the time modifier string AT "First::Meet::", determined the general resolving method for the tracing problem. If the above conditions and the relationship between data element size/proportion were satisfied, the corresponding resolving method for the relevant rumination framework could be invoked to generalize the solution method for such problems and verified the results.

The constraint relationship of semantic component's attributes and the size or proportion relationship of data elements could form another set of preconditions for rumination framework. The rumination framework also corresponded to the action plan such as problem scene identifiers, condition formula sets, and the general formula sets. Matching and replacing the scene identifiers and data element variable names of the rumination framework based on the actual situation of the problem; While satisfying the preconditions of rumination framework, the dynamic semantic circle of the new problem was generated. The principle of result verification was consistent with the previous methods and would not be repeated here.

4.2.1. Instance C

"There is a circular track with a circumference of 600 meters (C1), where A and B start simultaneously at the same

place (C2), and in the same direction (C3). A runs 300 meters per minute (C4), and B runs 400 meters per minute (C5). How many minutes do the two persons meet for the first time (C6)?"

After semantic annotating, semantic framework recognizing, global scene analyzing, data element variable naming, dynamic semantic circle (included general formulas and condition formulas) generating, commonsense knowledge referring, and thinking mechanism invoking, the semantic relationship rumination framework and corresponding semantic components could be obtained after verifying the resolving results.

The semantic relationship rumination framework was showed as follows:

RMA_C_TYPE4_2_1 /// Rumination framework ID

Circular::Track::Circumference, A::Speed, B::Speed, First~Time~::Meet::Time, ///input variables

sysFirstsys ///the scene identifier of the data element variables

Tracing::Distance, Tracing::Speed, Tracing::Time, /// new added middle data element variables;

4 ///condition formulas quantity

Existed, Tracing::Distance, /// antecedents

9 Circular::Track::Circumference, Tracing::Distance=-Left, Right, ///formula body

Existed, Tracing::Distance, /// antecedents

9 Tracing::Distance, Circular::Track::Circumference=-Left, Right, ///formula body

More~Than~, B::Speed, A::Speed, /// antecedents

6 B::Speed, Tracing::Speed, A::Speed=-Total, PartA, PartB,///formula body

More~Than~, A::Speed, B::Speed, /// antecedents

6 A::Speed, Tracing::Speed, B::Speed=-Total, PartA, PartB,///formula body

2 ///general formula quantity

9 Tracing::Time, First~Time~::Meet::Time=-Left, Right, ///formula body

9 First~Time~::Meet::Time, Tracing::Time=- Left, Right, ///formula body

The semantic component set (context constraint *SC4_2_1*) corresponding to the semantic relationship rumination pattern could be represented as follows:

SC4_2_1 ///semantic component set ID

RMA_C_TYPE 4_2_1 /// rumination framework ID

EN::AC::PL::AP::ER:: ///The semantic components string whose constraints could be relaxed.

AB Simultaneous~Departure~::Same~Starting~Place~:: Same~Direction~:: ///action modification constraint (string)

AC Run:: ///actions(string)

AE NULL ///subject modification (string), "NULL" represented none.

AO NULL ///object modification (string), "NULL" represented none.

AP Circular::, /// place modification (string).

AR NULL ///action mode (string), "NULL" represented none.

AT First~Time~::Meet::,///time modification (string).

ER Person:: ///Subject attributes(string), "NULL" represented none.

EN A::B:: ///Subjects (string)

PL Circular::Track:: /// places (string).

OB NULL ///objects (string), "NULL" represented none.

4.2.2. Instance D

"On a circular track with a circumference of 600 meters (C1), a truck and a car depart clockwise from the same starting place at the same time (C2). The truck runs 300 meters per minute (C3), and the car runs 400 meters per minute (C4). How many minutes do the two vehicles meet for the first time (C5)"

After semantic annotating, semantic framework recognizing, and global scene analyzing, the generated problem semantic components (contextual constraints) were compared with the corresponding information in Instance C. The corresponding semantic component approximation index was the sum of the level numbers of two concepts with the same attribute; The approximate index of the semantic relationship rumination framework was the sum of the approximate index of each replaceable semantic component (refer to Table 8).

Table 8. *Semantic components comparison.*

ID	Type	Instance D	Instance C	Index
1	EN	Truck::Car::	A::B::	8
2	AC	Run::	Run::	0
3	PL	Circular::Track::	Circular::Track::	0
4	AP	Circular::	Circular::	0*
5	ER	Vehicle::	Person::	2*
Total				8

Note: AP and ER were semantically included in PL and EN, and the indexes would not be calculated again in the total.

On the premise that the approximation index met the threshold, replaced the semantic components in the semantic rumination framework "*RMA_C_TYPE4_2_1*" of Instance C with the corresponding semantic components in Instance D, and generated the new rumination framework "*RMA_D_TYPE4_2_2*", as shown below:

RMA_D_TYPE4_2_2 /// Rumination framework ID

Circular::Track::Circumference, Truck::Speed, Car::Speed, First~Time~::Meet::Time, ///Input Variables

sysFirstsys ///the scene identifier of the data element variables

Tracing::Distance, Tracing::Speed, Tracing::Time, /// new added middle data element variables;

4 ///condition formulas quantity

Existed, Tracing::Distance, /// antecedents

9 Circular::Track::Circumference, Tracing::Distance=-Left, Right, ///formula body

Existed, Tracing::Distance, /// antecedents

9 Tracing::Distance, Circular::Track::Circumference=-Left, Right, ///formula body

More~Than~, Truck::Speed, Car::Speed, /// antecedents

6 Car::Speed, Tracing::Speed, Truck::Speed=-Total, PartA, PartB,///formula body

More~Than~, Truck::Speed, Car::Speed,

/// antecedents

6 Truck::Speed, Tracing::Speed, Car::Speed=-Total, PartA, PartB,///formula body

2 ///general formula quantity

9 Tracing::Time, First~Time~::Meet::Time=-Left, Right, ///formula body

9 First~Time~::Meet::Time, Tracing::Time=- Left, Right, ///formula body

The information of the generated rumination framework should be stored into the data element variable table and the dynamic semantic circle separately. After resolving Instance D by machine thinking mechanism and comparing the results with the general method, it could be confirmed that the above type of problems could be solved using the semantic relationship rumination framework.

4.3. Calculation Action Instance

Based on the deduction and calculation operations in action set such as "add/subtract/multiply/divide/rollback/…", enumerated and executed the operations, verified with the commonsense knowledge and result. The reasonable action sequence was recognized as the new common knowledge, and new theorems might be derived. This rumination computing mode had the possibility of achieving autonomous expansion for mathematic theory. Established the intelligent system to enumerate the possible actions and explore the new resolving method when the system was idle, observed, verified, and compared the system output until all possible resolving paths were explored and verified. In this rumination computing mode, all calculation actions were executed on the Intelligent Operating Environment (IOE). Relying on the IOE to achieve system state monitoring, probing backtracking, and robust calculation. The principles for result verification were consistent with the previous method and would not be repeated here.

Instance E: "A farmer raises 3000 chickens and rabbits (C1), totaling 7200 feet (C2). How many chickens and rabbits are there respectively (C3) ?"

4.3.1. Semantic Recognizing

After semantic framework matching and global semantic analysis, the following data element variable table (refer to Table 9) and local formula table (refer to Table 10) could be obtained:

Table 9. *Data element variables.*

No	Name	Type	Value
1	Chicken^Rabbit^::Total::Quantity	Known	3000
2	Chicken^Rabbit^::Total::Foot~Number~	Known	7200
3	Chicken::Quantity	question	?
4	Rabbit::Quantity	question	?
5	Chicken::Foot~Number~	Middle	?
6	Rabbit::Foot~Number~	Middle	?
7	Chicken::Foot~Number~::Quantity~::Rate	Constant	2
8	Rabbit::Foot~Number~::Quantity~::Rate	Constant	4

Table 10. *Local formula table.*

ID	Formula ID	Data Element Variable	Formula Variable
1	6	Chicken^Rabbit^::Total::Quantity, Chicken::Quantity, Rabbit::Quantity,	Total, PartA, PartB,
2	6	Chicken^Rabbit^::Total::Foot~Number~, Chicken::Foot~Number~, Rab-	Total, PartA, PartB,

ID	Formula ID	Data Element Variable	Formula Variable
		bit::Foot~Number~,	

4.3.2. Commonsense Knowledge

The commonsense calculation relationships implied in the context were shown in the Table 11 below:

Table 11. *Commonsense formulas.*

ID	Formula ID	Formula ID	Formula ID
1	3	Chicken::Foot~Number~, Chicken::Quantity, Chicken::Foot~Number~::Quantity~::Rate,	1_Vaviable,2_Vaviable, Times,
2	3	Rabbit::Foot~Number~, Rabbit::Quantity, Rabbit::Foot~Number~::Quantity~::Rate,	1_Vaviable,2_Vaviable, Times,

Commonsense knowledge about calculation actions included: the same property variables could perform addition or subtraction action; If there was a proportional relationship between two property variables, they could perform multiplication or division action; In the realm of elementary mathematics, the data elements corresponding to the physical world were all in the range of 0 and positive numbers.

1) First stage

At the beginning, based on the above commonsense and the 4 variables with the types of "Known" and "Constant" in Table 10, the possibly matched addition and subtraction actions included:

(1) "Chicken::Foot~Number~::Quantity~::Rate" + "Rabbit::Foot~Number~::Quantity~::Rate" (R1)

(2) "Chicken::Foot~Number~::Quantity~::Rate" - "Rabbit::Foot~Number~::Quantity~::Rate" (Negative value, eliminated based on commonsense knowledge)

(3) "Rabbit::Foot~Number~::Quantity~::Rate" - "Chicken::Foot~Number~::Quantity~::Rate" (R2)

Possibly matched multiplication and division actions included:

*(4) "Chicken^Rabbit^::Total::Quantity" * "Chicken::Foot~Number~::Quantity~::Rate" (R3)*

*(5) "Chicken^Rabbit^::Total::Quantity" * "Rabbit::Foot~Number~::Quantity~::Rate" (R4)*

(7) "Chicken^Rabbit^::Total::Foot~Number~" / "Chicken::Foot~Number~::Quantity~::Rate" (R5)

(8) "Chicken^Rabbit^::Total::Foot~Number~" / "Rabbit::Foot~Number~::Quantity~::Rate" (R6)

The six generated middle variables R1-R6 (physical meanings were not yet determined) were added to the known variables in the data element variable table. Checked if there were any middle variables that matched the problem result. If there were, the algorithm returned success and began the second result verification stage; If there weren't, the algorithm executed the above actions iteratively.

2) Second stage

Excluding the continuous use of the same variable in iterations, the possibly matched addition and subtraction actions include:

(8) (R3) - "Chicken^Rabbit^::Total::Quantity" (Negative value, eliminated based on commonsense knowledge)

(9) "Chicken^Rabbit^::Total::Quantity"- (R3) (R7)

(10) (R4) - "Chicken^Rabbit^::Total::Quantity" (R8)

(11) "Chicken^Rabbit^::Total::Quantity" – (R4) (Negative value, eliminated based on commonsense knowledge)

……

Possibly matched multiplication and division actions included:

(N) (R7) / (R2) (R9)

(N+1) (R8) / (R2) (R10)

……

4.3.3. Result Verification

The correct results of the question were as follows:

"Rabbit::Quantity"=600,

"Chicken::Quantity"=2400.

Equal to the values of the steps "R9" and "R10" in the previous section. Therefore, it could be inferred that the calculation and derivation processes of "R9" and "R10" were correct.

5. Conclusion

"Proposing hypotheses" and "verifying and promoting" were the important ways for scientific theoretical innovation and technological development. This method for machine rumination computing proposed in this article was based on the massive use cases, implemented the logic closed loop for innovation and was the important approach for automatic

optimization of knowledge engineering systems and autonomous creative thinking evolution of machine intelligence systems [39]. In the future, we will deepen the research on the formal representation for mathematic concept system and axiomatic system, and explore machine methods to improve mathematic theories, continuously accumulate commonsense, deepen research on redundant/inclusive knowledge and conflict resolution techniques, and try to construct the large-scale machine intelligence application system [40].

Abbreviations

ChatGPT	Artificial Intelligence Chatbot Program Developed by OpenAI in the United States
IOE	Intelligent Operating Environment
AC	Semantic Component: Actions (String)
AE	Semantic Component: Subject Modification (String)
AO	Semantic Component: Object Modification (String)
AP	Semantic Component: Place Modification (String)
AR	Semantic Component: Action Mode (String)
AT	Semantic Component: Time Modification (String)
ER	Semantic Component: Subject Attributes (String)
EN	Semantic Component: Subjects (String)
PL	Semantic Component: Places (String)
OB	Semantic Component: Objects (string)
AB	Semantic Component: Action Modification Constraint (String)
Ci, 1<=i<=n	Clause Identifiers
ID	Identifier

Author Contributions

Ping Zhu: Conceptualization, Methodology, Software, Writing – original draft

Pohua Lv: Project administration, Resources

Weiming Zou: Formal Analysis, Supervision

Xuetao Jiang: Investigation

Jin Shi: Data curation

Yang Zhang: Writing – review & editing

Yirong Ma: Validation

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] Damirchi R, Amini A. Non-Functional Requirement Extracting Methods for AI-Based systems: A Survey. International Conference on Computer and Knowledge Engineering. IEEE, 2023, pp. 535-539. https://doi.org/10.1109/ICCKE60553.2023.10326269

[2] Sari J E, Priyadi Y, Riskiana R R. Implementation of Semantic Textual Similarity Between Requirement Specification and Use Case Description Using WUP Method (Case study: Sipjabs application). IEEE World AI IoT Congress (AIIoT). IEEE, 2022, pp. 681-687. https://doi.org/10.1109/AIIoT54504.2022.9817311

[3] Sunita Chulani, Clay Williams, and Avi Yaeli. Software Development Governance and its Concerns. The 1st International Workshop on Software Development Governance (SDG '08). Association for Computing Machinery, New York, NY, USA, 2008, pp. 3–6. https://doi.org/10.1145/1370720.1370723

[4] Paul L. Bannerman. Software Development Governance: A Meta-Management Perspective. The 2009 ICSE Workshop on Software Development Governance (SDG '09). IEEE Computer Society, USA, 2009, pp. 3–8. https://doi.org/10.1109/SDG.2009.5071329

[5] Orit Hazzan and Yael Dubinsky. Can Diversity in Global Software Development be Enhanced by Agile Software Development. The 2006 International Workshop on Global Software Development for the Practitioner (GSD '06). Association for Computing Machinery, New York, NY, USA, 2006, pp. 58–61. https://doi.org/10.1145/1138506.1138520

[6] F. G. Dias, E. A. Schmitz, M. L. M. Campos, A. L. Correa, and A. J. Alencar. Elaboration of Use Case Specifications: An Approach Based on Use Case Fragments. The 2008 ACM Symposium on Applied Computing (SAC '08). Association for Computing Machinery, New York, NY, USA, pp. 614–618. https://doi.org/10.1145/1363686.1363835

[7] Ivar Jacobson, Ian Spence, and Brian Kerr. Use-Case 2.0. Commun. ACM, 2016, 59(5), pp. 61–69. https://doi.org/10.1145/2890778

[8] Wilson P. Paula Filho. Quality Gates in Use-Case Driven Development. The 2006 International Workshop on Software Quality (WoSQ '06). Association for Computing Machinery, New York, NY, USA, 2006, pp. 33–38. https://doi.org/10.1145/1137702.1137710

[9] Silverio Martínez-Fernández, Justus Bogner, Xavier Franch, Marc Oriol, Julien Siebert, Adam Trendowicz, Anna MariaVollmer, and Stefan Wagner. Software Engineering for AI-Based Systems: A Survey. ACM Trans. Softw. Eng. Methodol. 31, 2, Article 37e (April 2022), pp. 1-59. https://doi.org/10.1145/3487043

[10] Rob Pooley. Software Engineering and Performance: A Roadmap. The Conference on the Future of Software Engineering (ICSE '00). Association for Computing Machinery, New York, NY, USA, 2000, pp. 189–199. https://doi.org/10.1145/336512.336553

[11] A Mori. Anomaly Analyses to Guide Software Testing Activity. IEEE 13th International Conference on Software Testing, Validation and Verification (ICST), Porto, Portugal, 2020, pp. 427-429. https://doi.org/10.1109/ICST46399.2020.00055

[12] J. Joo, S. Yoo and M. Park, Poster: Test Case Prioritization Using Error Propagation Probability. IEEE 13th International Conference on Software Testing, Validation and Verification (ICST), Porto, Portugal, 2020, pp. 398-401. https://doi.org/10.1109/ICST46399.2020.00047

[13] Thomas R N, Roopam G, A Survey on Machine Learning Approaches and Its Techniques. International Students' Conference on Electrical, Electronics and Computer Science. IEEE, 2020, pp. 1-6. https://doi.org/10.1109/SCEECS48394.2020.190

[14] Radhya S, Syahfiera Tasik A M, Sabran M F, et al. Systematic Literature Review: Machine Learning in Education to Predict Student Performance. International Conference on Electrical and Information Technology. IEEE, 2022, pp. 350-356. https://doi.org/10.1109/IEIT56384.2022.9967900

[15] Cao Y. Design and Implementation of an Intelligent Machine Learning System Based on Artificial Intelligence Computing. International Conference on Data Analytics, Computing and Artificial Intelligence. IEEE, 2023, pp. 707-711. https://doi.org/10.1109/ICDACAI59742.2023.00141

[16] Abdulaziz E, Aleryani A. A Survey on Big Data Analytics for Education. International Conference on Emerging Smart Technologies and Applications. IEEE, 2022, pp. 1-6. https://doi.org/10.1109/eSmarTA56775.2022.9935459

[17] Vaidya M G, Kshirsagar M M. A Survey of Algorithms, Technologies and Issues in Big Data Analytics and Applications. International Conference on Intelligent Computing and Control Systems. IEEE, 2020, pp. 347-350. https://doi.org/10.1109/ICICCS48265.2020.9121064

[18] H. L. Zhang, J. Liu, T. Li, Y. Xue, S. Xu and J. Chen, Extracting Sample Data Based on Poisson Distribution. International Conference on Machine Learning and Cybernetics (ICMLC), Ningbo, China, 2017, pp. 374-378. https://doi.org/10.1109/ICMLC.2017.8108950

[19] Gandhi J, Gandhi P, Gosar A, et al. Natural Language Processing-Based Math Word Problem Solver and Synoptic Generator. International Conference on Electronics and Sustainable Communication Systems. IEEE, 2022, pp. 12-16. https://doi.org/10.1109/ICESC54411.2022.9885451

[20] Mandal S, Naskar K S. Classifying and Solving Arithmetic Math Word Problems—An Intelligent Math Solver. IEEE Transactions on Learning Technologies, 2021, 14(1), pp. 28-41. https://doi.org/10.1109/TLT.2021.3057805

[21] Kotwal H, Patnaik K G. Solving Arithmetic English Word Problems Using Intermediate Representation. International Conference on Computing, Communication, Control and Automation. IEEE, 2023, pp. 1-5 https://doi.org/10.1109/ICCUBEA58933.2023.10391982

[22] Meng H, Yang T, Yu X. A Bi-Channel Math Word Problem Solver with Understanding and Reasoning. International Conference on Engineering, Technology & Education. IEEE, 2021, pp. 29-34. https://doi.org/10.1109/TALE52509.2021.9678542

[23] A. Tendle, A. Little, S. Scott and M. Rashedul Hasan, Self-Supervised Learning in the Twilight of Noisy Real-World Datasets. 21st IEEE International Conference on Machine Learning and Applications (ICMLA), Nassau, Bahamas, 2022, pp. 461-464. https://doi.org/10.1109/ICMLA55696.2022.00074

[24] S. Kumar, A. Phukan and A. Sur, IPCL: Iterative Pseudo-Supervised Contrastive Learning to Improve Self-Supervised Feature Representation. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Seoul, Korea, 2024, pp. 6270-6274. https://doi.org/10.1109/ICASSP48485.2024.10447607

[25] T. -P. Nguyen, S. Razniewski, J. Romero and G. Weikum, Refined Commonsense Knowledge From Large-Scale Web Contents. IEEE Transactions on Knowledge and Data Engineering, 2023, 35(8): 8431-8447. https://doi.org/10.1109/TKDE.2022.3206505

[26] Y. Shi, K. Wu and M. Zhang, COVID-19 Pandemic Trend Prediction in America Using ARIMA Model. International Conference on Big Data, Information and Computer Network (BDICN), Sanya, China, 2022, pp. 72-79. https://doi.org/10.1109/BDICN55575.2022.00022

[27] R. Saegusa, G. Metta, G. Sandini and L. Natale, Developmental Perception of the Self and Action. IEEE Transactions on Neural Networks and Learning Systems, 2014, 25(1): 183-202. https://doi.org/10.1109/TNNLS.2013.2271793

[28] S. Salloum, J. Z. Huang and Y. He, Random Sample Partition: A Distributed Data Model for Big Data Analysis. IEEE Transactions on Industrial Informatics, 2019, 15(11): 5846-5854. https://doi.org/10.1109/TII.2019.2912723

[29] X. Wang, L. T. Yang, H. Liu and M. J. Deen, A Big Data-as-a-Service Framework: State-of-the-Art and Perspectives. IEEE Transactions on Big Data, 2018, 4(3): 325-340. https://doi.org/10.1109/TBDATA.2017.2757942

[30] S. Sarker, M. S. Arefin, M. Kowsher, T. Bhuiyan, P. K. Dhar and O. J. Kwon. A Comprehensive Review on Big Data for Industries: Challenges and Opportunities. IEEE Access, 2023, 11(0): 744-769. https://doi.org/10.1109/ACCESS.2022.3232526

[31] Ping Zhu, Pohua Lv, Jin Shi, Xuetao Jiang, Weiming Zou, and Yirong Ma. Design and implementation of text understanding system based on semantic tagging instances. Proceedings of the 2023 4th International Conference on Artificial Intelligence in Electronics Engineering (AIEE '23). Association for Computing Machinery, New York, NY, USA, pp. 108–116. https://doi.org/10.1145/3586185.3586190

[32] Zenkert J, Fathi M. Multidimensional Knowledge Representation of Text Analytics Results in Knowledge Bases. International Conference on Electro Information Technology. IEEE, 2016, pp. 541-546. https://doi.org/10.1109/EIT.2016.7535297

[33] Thiombiano J, Traore Y, Malo S, et al. Semantic Annotation of Resources Based on Ontologies: Application to a Knowledge Sharing Platform on Meningitis. International Conference on Smart Cities and Communities. IEEE, 2020, pp. 1-6. https://doi.org/10.1109/SCCIC51516.2020.9377332

[34] Son H, Weiland J. Lightweight Semantic Segmentation Network for Semantic Scene Understanding on Low-Compute Devices. International Conference on Intelligent Robots and Systems. IEEE, 2023, pp. 62-69. https://doi.org/10.1109/IROS55552.2023.10342110

[35] Deleep Kumar S, Sunitha C, Ganesh A. Semantic Representation of Texts in Indian Languages — A Review. International Conference on Inventive Systems and Control. IEEE, 2018, pp. 40-42. https://doi.org/10.1109/ICISC.2018.8399119

[36] Zhu P, Lv P H, Shi J, Jiang X T, et al. Semantic Inheritance and Overloading. International Conference on Software Engineering and Artificial Intelligence. IEEE, 2022, pp. 1-9. https://doi.org/10.1109/SEAI55746.2022.9832076

[37] Hu S, Wang Z, Zhang S, et al. Foreign Fibers Detection Using Improved Otsu-Based Maximum Entropy Algorithm in Spinning Process. International Conference on Robotics and Automation Sciences. IEEE, 2022, pp. 206-210. https://doi.org/10.1109/ICRAS55217.2022.9842190

[38] Alahmed Y, Abadla R, Badri A A, et al. "How Does ChatGPT Work" Examining Functionality to the Creative AI ChatGPT on X's (Twitter) Platform. International Conference on Social Networks Analysis, Management and Security. IEEE, 2023, pp. 1-7. https://doi.org/10.1109/SNAMS60348.2023.10375450

[39] Alkhatib J O. An Effective Assessment Method of Higher Order Thinking Skills (Problem-Solving, Critical Thinking, Creative Thinking, and Decision-Making) in Engineering and Humanities. Advances in Science and Engineering Technology International Conferences. IEEE, 2022, pp. 1-6. https://doi.org/10.1109/ASET53988.2022.9734856

[40] Zhu Ping, Lv Pohua, Zou Weiming, Jiang Xuetao, Shi Jin, Zhang Yang, Ma Yirong. Construction of Super Large Interpretable Machine Intelligence System. Computer Technology and Development, November 2024. (To be published) https://doi.org/10.20165/j.cnki.ISSN1673-629X.2024.0232

Biography

Ping Zhu PHD, engineering professor. Has been engaged in intelligent system R&D more than 26 years. He has successively undertaken the application, R&D, and management of China 863 plan smart city (phase II project)'s task; He had assisted the Beijing municipal commission of economy and information technology in the preparation of smart city planning of Beijing sub center and obtained the Commendation as the Deputy group leader; More than 50 scientific and technological papers have been published by the first author at home and abroad. As the session chair, he participated in the CCEAI 2021 and CAIT 2020 international academic conferences, served as the technical committee member of international academic conferences for many times. He is member of Hong Kong society of Robotics and Automation (HKSRA), Asia-Pacific Institute of Science and Engineering (APISE) and International Association of Science and Engineering Development (IASED).

Research Field

Ping Zhu: semantic understanding, thinking machine, complex algorithm designing.

Pohua Lv: smart city, big data, semantic communication.

Weiming Zou: big data, intelligent engineering.

Xuetao Jiang: digital government, intelligent engineering.

Jin Shi: smart city, data model.

Yang Zhang: big data.

Yirong Ma: big data, data model.