Reasoning Method on Knowledge about Functions and Operators

Nhon V. Do
Faculty of Computer Science
University of Information Technology, VNU-HCM
Ho Chi Minh City, Vietnam

Hien D. Nguyen
Faculty of Computer Science
University of Information Technology, VNU-HCM
Ho Chi Minh City, Vietnam

Thanh T. Mai
Faculty of Information Technology
Binh Duong University
Binh Duong Province, Vietnam

***Abstract*—In artificial intelligence, there are many methods for knowledge representation. One of the effective models is the Computational Object Knowledge Base model (COKB model), which can be used to represent the total knowledge and to design the knowledge base component of practical intelligent systems. Besides, reasoning methods also play an important role in knowledge base systems. In fact, a popular form of knowledge domain is knowledge about function and operations. These knowledge domains have many practical applications, especially in educational applications, such as Solid Geometry, Analytic Geometry. However, the current methods cannot reason on knowledge about Function and Operators. In this paper, we will present a reasoning method to solve problems on COKB model. These problems are related to knowledge about Functions and Operators. Also, this method has been applied to design some intelligent systems in education. Using this reasoning method, systems can solve problems in some educational knowledge domains automatically with their solutions are step-by-step.**

Keywords*—knowledge representation; knowledge based system; intelligent problem solver; automated reasoning

I. Introduction

Intelligent problem solvers (IPS), presented in [1], can consist of components such as theorem proving, inference engines, search engines, learning programs, classification tools, statistical tools, question-answering systems, machine translation systems, knowledge acquisition tools. The IPS system can solve problems in general forms. Users only declare hypothesis and goal of problems base on a simple language but strong enough for specifying problems. The hypothesis can consist of objects, relations between objects or between attributes. It can also contain formulas, determination properties of some attributes or their values. The goal can be to compute an attribute, to determine an object, a relation or a formula. After specifying a problem, users can request the program to solve it automatically or to give instructions that help them to solve it themselves.

An important component in IPS system is knowledge base. This component contains knowledge to solve problems of the system. Knowledge base is designed by knowledge representation methods. Besides the knowledge base, these methods also should be convenient for designing inference engine and interface of the system. Some methods for representation knowledge have been studied. Many classical methods to design knowledge model has been proposed and apply, such as logic, conceptual graph [2], [3], etc. But those results are not efficient for executing and using in the real application.

In real application, a popular component of knowledge domain is knowledge component about operations. Operators between objects of knowledge domain are necessary concepts; they help for representation this knowledge exactly. Besides operators component, knowledge domains can contain relations as function between objects. Example in Knowledge domain about 2D-Analytic Geometry, it contains not only operators between vectors, but also functions about *distance* between two points, between a point and a line, etc. However, the current knowledge models are not effectively for representing knowledge about operators and function. These components cannot be separated but they have to be considered in a knowledge-based system that has many components, this system also includes concepts and relations.

In this paper, Functions and Operators components in model of Computational Objects Knowledge Bases (COKB) will be researched. After that, model of problems will be built and algorithms for reasoning to solve them will be also designed. Moreover, the IPS systems have been constructed; it can solve problems in some educational knowledge domains automatically. The solutions of these systems are naturally, step-by-step, like human's thinking.

The next section will give an overview of the related work. Section III will present a knowledge model about Functions and Operators; this model is constructed based on COKB model. In section IV, problems on this model will be modeled and classified, and algorithms to solve them also will be designed. Based on knowledge model and algorithms in previous sections, the systems of Intelligent Problem Solvers will be constructed in section V. These systems have been applied in some knowledge domains about physics and mathematics. In the last section, we will present conclusion of this study and future work.

II. Related Work

An algebraic structure of elementary knowledge has been presented base on first order logic, as in [4]. This study have solved problem about information equivalent of knowledge. In the other research, in [5], this problem have also solved by constructing automorphic of knowledge-based. However, knowledge domains in these studies are very simple.

Some methods for automated theorem proving has been researched, such as using Groebner bases [6] or algebraic structure based on Temporal logics [7], etc. However, those methods cannot represent knowledge domain precisely. Moreover they cannot implement the way of human's thinking to solve problems.

A modern approach for representing knowledge is ontology. Ontology has been researched and developed in practice application [8, 9, 10]. Ontology COKB has been used in designing some intelligent systems in education (see [11, 12]). In the other study, authors built an ontology for some kinds of relations based on predicate logic and description logic [13], however, this model have drawback to represent complexity relations in human's knowledge.

The study in [14] presents algebraic structure of knowledge about relations; thought that, authors represent a method for knowledge acquisition. However, this model has not yet mentioned structure of concepts and rules.

In fact, a popular form of knowledge domain is knowledge about function and operations. These knowledge domains have many practical applications, especially in educational applications. However, the current methods cannot reason on knowledge about Function and Operators. In [15], authors presented a method for representation this knowledge, but this model is still independence with knowledge about operators and function.

Besides, problems need to be modeled so that we can design algorithms for solving problems automatically. For automatic inference methods, the most important thing is controlling strategy for generating new facts from known facts. Many automatic inference techniques have been studied quite completely in a general level, including: (see [3, 16])

1) Unification routine in methodology for knowledge base representation by first-order logic.

2) Inference by forward chaining

3) Inference by backward chaining

4) Combining forward and backward chaining together with using heuristic rules.

However, the above results are still too general. Some models and techniques are still partial and not good enough for constructing a knowledge system with requirements which are not easy for designer to set up and implement.

In [12, 17], authors presented a reasoning method using Sample Problems. In the processing of solving a practical problem, this method will search for relating problems which were solved before and then apply it to find the solution of the current problem. However, in these models, components about Operators and Functions have not yet researched completely. Some problems about them have not yet been solved, such as: specification of operators and functions, compute an expression between objects, determine value of a function, etc.

UMS Software, in [22], is a program that supports to solve algebraic problems. Some kinds of problem can be solved by this program: solve the equation and basic inequalities, solve the system of equation in two variables, factorization of a Polynomial, simplification of algebraic expression, radical simplification, etc. Besides the solution by textual, UMS Software accompanies the solution with voice comments. But the program is not equipped a knowledge base of algebraic knowledge, it solve problems as frame. The other program in [23] is a website for supporting to solve problems automatically in mathematical knowledge domains. It shows the solutions step-by-step. However, especially in geometry, objects in this program are still simple, and it has limit about number of objects in problems (five objects), so specification about relations between objects is not completely.

III. Knowledge Model about Functions and Operators

A. Model of Computational Object Knowledge Base

The model of Computational Objects Knowledge Bases (COKB) has been established from the integration of ontology engineering, object-oriented modeling and symbolic computation programming. This way also gives us a method to model problems and to design algorithms. The models are very useful for constructing components and the whole knowledge base of knowledge-based systems in practice.

Definition 2.1: The model of Computational Object Knowledge Base (COKB) consists of six components:

(C, H, R, Ops, Funcs, Rules)

The meanings of the components are as follows:

- **C** is a set of concepts of computational objects.
- **H** is a set of hierarchy relation (IS-A relation) on the concepts.
- **R** is a set of relations between the concepts.
- **Ops** is a set of operators.
- **Funcs** is a set of functions.
- **Rules** is a set of rules.

Each concept in C is a class of Com-objects. The structure Com-Objects can be modeled by **(*Attrs, F, Facts, RulesObj*)**. *Attrs* is a set of attributes, *F* is a set of equations called computation relations, *Facts* is a set of properties or events of objects, and *RulesObj* is a set of deductive rules on facts.

H represents these special relations on *C*. This relation is an ordered relation on the set C, and *H* can be considered as the Hasse diagram for that relation. *R* is a set of other relations on *C*, and in case a relation r is a binary relation it may have properties such as reflexivity, symmetry, etc.

The set *Ops* consists of *operators* on *C*. This component represents a part of knowledge about operations on the objects. Almost knowledge domains have a component consisting of operators. The set *Funcs* consists of functions on Com-Objects. Knowledge about functions is also a popular kind of knowledge in almost knowledge domains in practice, especially fields of natural sciences such as fields of mathematics, fields of physics.

The set *Rules* represents for deductive rules. The rules represent for statements, theorems, principles, formulas, and so forth. Almost rules can be written like the form "if <facts> then <facts>". In the structure of a deductive rule, <facts> is a set of facts with certain classification. Facts must be classified so that the knowledge component *Rules* can be specified and processed in the inference engine of intelligent systems.

B. Knowledge about Functions and Operators

1) Knowledge about Functions

There is wide knowledge and many problems related to the functional component in real knowledge domains, such as knowledge of plane geometry, solid geometry, knowledge of alternating current in physics. For example, intersection between two planes is a line, so Intersection can be a function; the common perpendicular of two skew lines returns a line can be modeled as a function. So that, Funcs component in COKB model is necessary to describe functional knowledge in reality. In [18], authors have researched Funcs component. However, some problems on Functional knowledge have not been mentioned, such as specification of functions, determine value of function.

Structure of a function as followed:

Definitions of functions – form 1:

```
function-def ::= FUNCTION name;
                  ARGUMENT: argument-def+
                  RETURN: return-def;
                  [constraint]
                  [facts]
          ENDFUNCTION;
```

Definitions of functions – form 2:

```
function-def ::= FUNCTION name;
                 ARGUMENT: argument-def+
                 RETURN: return-def;
                 [constraint]
                 [variables]
                 [statements]
         ENDFUNCTION;
 argument-def   ::= name, <name>: type
 return-def     ::= name : type
statements      ::= statement-def+
statement-def   ::= assign-stmt | if-stmt | for-stmt
asign-stmt      ::= name := expr;
if-stmt         ::= IF logic-expr THEN statements+
                      ENDIF; | IF logic-expr THEN
                      statements+ ELSE statements+
                      ENDIF;
 for-stmt       ::=  FOR name IN [range] DO
                statements+ ENDFOR;
```

Eg. 2.1: In Knowledge domain about Two dimension analytical geometry, DISTANCE function to compute distance between two point has been defined like this:

```
FUNCTION   <DISTANCE >  :
      ARGUMENT: A::POINT, B::POINT
      RETURN: d::REAL
      constraint:
  d:=$\sqrt{(B.x-A.x)^2+(B.y-A.y)^2}$
ENFUNCTION
```

2) Knowledge about Operators

In knowledge domains about computation, Ops component is very important to describe them. This component represents operators between objects in knowledge domain. On the basis of operators, knowledge is equipped efficient computational methods for knowledge processing. An operator in this model is a binary mapping: Ci × Ci→ Ci, with Ci is a concept in set C [16]. Structure of an operator as followed:

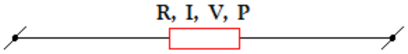
```
operator-def ::= OPERATOR <name>
                  ARGUMENT: argument-def+
                  RETURN: return-def;
                  PROPERTY: prob-type;
                ENDOPERATOR
argument-def ::= name, <name>: type
return-def ::= name : type
prob-type ::= commutative | associative | identity
```

Definition 2.2: Expression of objects is defined like this:

expr ::= object | expr <operator> expr

Rules for determining an operator: Rules-set of Ops-model has been classified two kinds: kind for determining an operator and kind for inference rules of knowledge. The result of an operator is an object of concept in set C. The values of attributes of this object are determined based on rules for determining operator in Rules-set.

Eg. 2.2: In Knowledge domain about Direct Current Electrical Circuit, CIRCUIT concept is modeled by Com-Object, it includes:



Attrs = {R,V,I,P}: set of attributes of CIRCUIT
R,V,I,P: values of resistance, potential difference, current flow, power of circuit.

$$F=\left\{I=\frac{V}{R},P=V.I,P=\frac{V^2}{R},P=I^2R\right\}$$

RulesObj = { }

Operator " +" performs the series connection between two circuits has been represented like this:

```
OPERATOR  <  +  > :
      ARGUMENT: CIRCUIT, CIRCUIT
      RETURN: CIRCUIT
      PROPERTY: commutative, associative
  ENDOPERATOR
```

Result of this operator has been determined base on rule R1 in section V.A

C. Basic Techniques

1) Classify kinds of facts

In the COKB model, there are 12 kinds of facts accepted. These kinds of facts have been proposed from the researching on real requirements and problems in different knowledge domains. The kinds of facts are as follows:

- **Fact of kind 1**: proposition state information about object kind or type of an object.
- **Fact of kind 2**: a proposition states determination of an

object or an attribute of an object.

- **Fact of kind 3**: a proposition states determination of an object or an attribute of an object by a value or a constant expression.
- **Fact of kind 4**: a proposition states equality on objects or attributes of objects.
- **Fact of kind 5**: a proposition states dependence of an object on other objects by a general equation.
- **Fact of kind 6**: a proposition states a relation between objects or attributes of the objects.
- **Fact of kind 7**: a proposition states determination of a function.
- **Fact of kind 8**: a proposition states determination of a function by a value or a constant expression.
- **Fact of kind 9**: a proposition states equality between an object and a function.
- **Fact of kind 10**: a proposition states equality between a function and another function.
- **Fact of kind 11**: a proposition states dependence of a function on other functions or other objects by an equation.
- **Fact of kind 12**: a proposition states a relation between functions.

The last six kinds of facts are related to knowledge about functions, the component **Funcs** in model COKB. The problem below gives some examples for facts related to functions. Problem: Let d be the line with the equation 3x + 4y - 12 = 0. P and Q are intersection points of d and the axes Ox, Oy.

a) Find the central point of PQ

b) Find the projection of O onto the line d.

For each line segment, there exists one and only one point that is the central point of that segment. Therefore, there is a function MIDPOINT(A, B) that outputs the central point M of the line segment AB. Part (a) of the above problem can be represented as to find the point I such that I = MIDPOINT(P,Q), a fact of kind 9. The projection can also be represented by the function PROJECTION(M, d) that outputs the projection point N of point M onto line d. Part (b) of the above problem can be represented as to find the point A such that A = PROJECTION(O,d), which is also a fact of kind 9.

2) Unification of facts

Unification algorithms of facts were designed and used in different applications such as the system that supports studying knowledge and solving analytic geometry problems, the program for studying and solving problems in Plane Geometry. The basic technique for designing deductive algorithms is the unification of facts. Based on the kinds of facts and their structures, there will be criteria for unification proposed. Then it produces algorithms to check the unification of two facts. For instance, when we have two facts *fact1* and *fact2* of kind 1-6, the unification definition of them is as follows: fact1 and fact2 are unified if they satisfy the following conditions

(1) *fact1* and *fact2* have the same kind k, and

(2) *fact1* = *fact2* if k = 1, 2, 6.
[*fact1*[1], {*fact1*[2..nops(*fact1*)]}] =
[*fact2*[1], {*fact2*[2..nops(*fact2*)]}] if k = 6
and the relation in *fact1* is symmetric.
lhs(*fact1*) = lhs(*fact2*) and
compute(rhs(*fact1*)) = compute(rhs(*fact2*)) if k =3.
(lhs(*fact1*) = lhs(*fact2*) and rhs(*fact1*) = rhs(*fact2*)) or
(lhs(*fact1*) = rhs(*fact2*) and rhs(*fact1*) = lhs(*fact2*)) if k = 4.
evalb(simplify(expand(lhs(*fact1*) - rhs(*fact1*) - lhs(*fact2*) + rhs(*fact2*))) = 0)
or evalb(simplify(expand(lhs(*fact1*) - rhs(*fact1*) + lhs(*fact2*) - rhs(*fact2*))) = 0) if k = 5.

IV. Problems and Reasoning Method

A. Model of Problems

Definition 3.1: Let knowledge K = (C, H, R, Ops, Funcs, Rules) as COKB model, model of problem include three sets as followed:

(O, F, G)

In which:

$O = \{O_1, O_2, \ldots, O_n\}$;
$F = \{f_1, f_2, \ldots, f_m\}$;
$G = \{ g_1, g_2, \ldots, g_m \}$

In the above model the set **O** consists of objects, **F** is the set of facts given on the objects, and **G** consists of goals.

The problem will be denoted by **(O, F) → G**

Eg. 3.1: **(**Knowledge domain about Solid geometry)

Problem: Let S.ABC be a triangular pyramid, and I, K be arbitrary points of edges SA, SC respectively such that IK is not parallel to AC. Determine the point of intersection of IK and plane (ABC).

The problem can be modeled by some facts below:

O = {POINT: S, A, B, C, I, K,
Triangle Pyramid: S.ABC}
F = {I belongs to SA, K belongs to SC,
IK is not parallel to AC},
G = {Determine: Point of Intersection of IK and Plane[ABC]}.

Besides that, the goal G in model of problem may be:

1) Determine an object or an attribute (or some attributes) of Object or compute a value of a function relative to objects.

2) Determine an object or an attribute (or some attributes) of Object or compute a value of a function relative to objects and adding some condition.

3) Arguments about relationship between objects according to parameters.

Moreover, each goal of problem on COKB which has presented above is a different kind, so to solve generality problem, we have classified the problem above to three kinds of problems (three class of problems). And each kind of problem will be presented below:

Definition 3.2: Give problem on COKB model has the goal G, with G is determining an object or an attribute (or some attributes) of Object or compute a value of a function relative to objects and *adding some condition*. This problem has the form:

$$(O,F) + (O_{condition}, F_{condition}) \rightarrow G$$

where ($O_{condition}$, $F_{condition}$) is condition (to be added) to determine the goal, that G has to ensure, denote H + $H_{condition}$ → G, in which: H is the hypothesis of the problem, $H_{condition}$ is condition that to G must be guaranteed.

Eg. 3.2: **(**Knowledge domain about 2D-analytic geometry)

Problem: Give two lines d_1:$x + y + 1 = 0$, d_2: $2x - y - 1 = 0$. Write a equation of a line (d), known (d) through M, with M point has coordinates (1;-1), and (d1) intersect with (d_1) and (d_2) at A, B with condition is $2\overrightarrow{MA} + \overrightarrow{MB} = \vec{0}$.

We have some facts below:

O = {POINT: M, A, B, LINE: d1,d2,d},
F = {equation of d1: x+y+1=0,
equation of d2: 2x -y -1=0,
M(1;-1),
d through M,
d intersect d1 and d2 at A, B},
$O_{condition}$ ={},
$F_{condition}$ = $\{2\overrightarrow{MA} + \overrightarrow{MB} = \vec{0}\}$,
G = {WRITE:equation of d}

Definition 3.3: Give problem on COKB model has the goal G, which is an argument about relationship between objects according to parameters. This problem has the form:

$$(O, F) + PAR \rightarrow G$$

where, PAR is a set of parameter and goal G is an argument about relationship between objects according to parameters.

Eg. 3.3: **(**Knowledge domain about 2D-analytic geometry)

Problem: Give a circle (C), equations of (C) is $(x-2)^2 + (y-3)^2 = 4$, and give A, B are two Point, A has coordinates (4;2), B has coordinates (0,m), with m is parameters of problem. Determine m so that AB cut (C).

O = {POINT:A, B, CIRCLE:C},
F = {equation of (C): $(x-2)^2 + (y-3)^2 = 4$,
A(4;2), B(0,m)},
G = {Determine: m so that AB cut (C)}.

B. Reasoning for problem solving:

Definition 3.4: Give a problem (O, F) → G on model COKB, and the goal G is to determine (or compute) attributes; M is the set of attributes considered in the problem, A ⊆ M. Denote L be the set of facts in the hypothesis.

- Each f ∈ Rules, each O_i ∈ O, each fun ∈ *Funcs*, each ops ∈ *Ops*, each eq_i ∈ *Equations* we define:

f(A) = A ∪ M_A(f), with M_A(f) is set of facts can be deduced from A by f.

O_i(A) = A ∪ $M_A(O_i)$, with $M_A(O_i)$ is set of facts deduced from A by O_i

fun(A) = A ∪ M_A(fun), with M_A(fun) is set of facts deduced from A by fun.

ops(A) = A ∪ M_A(ops), with M_A(ops) is set of facts deduced from A by ops.

eq_i(A) = A ∪ $M_A(eq_i)$, with $M_A(eq_i)$ is set of facts deduced from A by eq_i.

Each eq_j ∈ *Equations* | ($eq_i \neq eq_j$) :

$eq_i(A),eq_j(A)$ = A ∪ $M_A(eq_i,eq_j)$, with $M_A(eq_i,eq_j)$ is set of facts deduced from A by (eq_i,eq_j).

a) Suppose D = [d_1,d_2, …, d_m] is a list of elements, which d_j ∈ *Rules* or d_j ∈ *O* or d_j ∈ *Ops* or d_j ∈ *Funcs* or d_j ∈ *Equations*, that used to solve problem. Denote:

F_0 = F, F_1 = $d_1(F_0)$, F_2 = $d_2(F_1)$,..,F_m = d_m(F_m-1) and D(F) = F_m.

A problem is called *solvable* if there is a list D such that G ⊆ D(L).

The process from state F_i (by applied d_{i+1}) to state F_{i+1} is called a *reasoning step* (i=$\overline{0 \dots m-1}$).

b) Suppose a problem (O, F) → G is *solvable* and called S = [$step_1$, $step_2$,…, $step_m$] is list step of solution of problem. Each $step_i$ is *a step of solution* (i=1..m), it has the form:

$$(r, kf, nf)$$

r ∈ D: is rule can be applied to produce new facts or new objects, kf is set of known facts, which can be applied in r, nf is set of new facts, which was produced by r(kf).

When dealing with a practical problem, a convenient way to proceed is considering whether we have met a similar or related problem before or not. If so, then the solution for the problem can be obtained effectively. Or we determine the result of relating problems can be used to solve the practical problem. The related problem like this is called Sample Problem [12].

Definition 3.6: Give S is sample problem in Knowledge K of the form COKB model, consists of two parts:

$$(Psp, Ssp)$$

Psp is problem of S, it has form (Op, Fp) → Gp (the model of problem on COKB). Ssp is a list solution-step (by *definition 3.4*) , it has the structure Ssp = [$step_{1,}$ $step_{2,}$ $step_{3,\dots,}$$step_k$]. And we call facts(S) is set of hypothesis of S and goal(S) is set of goal of S, where facts(S) = {Op} ∪ {Fp}, goal(S) = {Gp}.

Problem P on COKB model is classified to three kinds (definition 3.1-3.3). This problem can be solved by using forward reasoning with heuristic rules to select rules or sample problems. There are alogithms to solve kinds of problem as followed:

Algorithm 3.1: Give a problem P1 = (O, F) → G on COKB model as definition 3.1, denote: H → G, *Problem P1 on COKB domain has been found by the following steps*:

- Step 1: Record the element in hypothesis and goal part
- Step 2: Check the goal G, if G is obtained the go to step 7.
- Step 3: Using heuristic rules to select a rule for producing new facts or new objects. If it has found a rule can be applied then go to step 2.
- Step 4: Using heuristic rules to select a sample problem for producing new facts or new objects. If it has found a sample problem can be applied then go to step 2.
- Step 5: Searching for any rule, which can be used to deduce new facts or new objects. If has found a rule can be applied then go to step 2.
- Step 6: Giving conclusion: Solution not found; stop the program.
- Step 7: Reduce the solution found by excluding redundant rules and information in the solution.

In deductive process to find rule can be applied for problem on the COKB, at *algorithm 3.1* we have used some heuristic rules to select rule below:

- Priority use of rules for determining objects.
- Transform objects to objects at a higher level in the hierarchical graph if there are enough facts.
- Use rules for producing new objects that contains elements, which are not in existing objects.
- Use rules for producing new objects that have relationship with existing objects, especially the goal, in necessary situations.
- Try to use deduction rules to get new facts, especially facts that have relationship with the goal.
- If we could not produce new facts or new objects then we should use parameters and equations.There are always new facts (relations or expressions) when we produce new objects.

Algorithm 3.2: The basic procedure for test a sample problem if can be applied.

Giving problem P = (O, F) → G on knowledge K of the form COKB model. Suppose Known_Facts is the set of known fact, which was saved during deduction process to solve problem P, and giving Rules_Sample is the set of sample problem in Knowledge K, have the following structure:

Known_Facts = {kf$_1$, kf$_2$, kf$_3$,…,kf$_n$}, each kf$_i$ is one of twelve kind of facts in K (i=1..n).

Rules_Sample = {sp$_1$, sp$_2$,…,sp$_m$}, each sp$_i$ is a sample problem in knowledge K (i=1…m).

Giving S as definition 3.6 is a sample problem in Knowledge K of the form COKB model (S ∈ Rules_Sample).

S can be applied on problem P has been found by the following procedure:

```
KF ← Known_Facts;
$check \leftarrow false$;
if facts(S) $\subseteq$ KF and goal(S) $\not\subset$ KF then
  $check \leftarrow true$;
end if;
if $check=true$ then
      S is a sample problem can be applied on P to
      produce news facts; return true;
else
      There is no sample problem found; return false;
end if;
```

⊆: is symbol of subset, in these cases, it was understood by unification of facts. facts(S) ⊆ KF is return true when facts(S) and KF can be unified.

Algorithm 3.3: With problem P on COKB model, the sample problem can be applied on P has been found by the following procedure: (*heuristics was used for this action*)

```
Sample_found ← $flase$;
SP ← Rules_Sample;
$find$ ← Use Heuristic rules to select S in SP, and S can
be applied on P (algorithm 3.2)
if $find = true$ then
      Sample_found ← $true$;
      break;
end if;
if Sample_found then
      S is a sample problem can be applied on P to
      produce news facts;
else
      S is a sample problem cannot be applied on P to
  produce news facts;
end if;
```

To find a sample problem in set of sample problem to resolve for problem on the COKB, at *algorithm 3.3* we have used some heuristic rules to select a sample problem below:

- Priority use of sample have used with high frequency.
- Priority use of sample for determining objects.
- Priority use of sample for producing new objects that have relationship with existing goal of problem.

Algorithm 3.4: Give a problem P2 on COKB has the form (O,F) + (O$_{condition}$, F$_{condition}$) → G as definition 3.2, denote H + H$_{condition}$ → G. Problem P2 can be resolved by applying algorithm 3.1, and priority using heuristic rules to select sample and rule below:

- Priority use of sample has hypothesis in set of condition of problem.
- Priority use of sample has the goals which have relationship with existing objects in set of condition of problem.

- Priority use of rules has hypothesis in set of condition of problem.
- Priority use of rules for determining objects or fact which have relationship with existing objects in set of condition of problem.

Algorithm 3.5: Give problem P3 on COKB model has the form (O,F) + PAR → G (by *definition 3.3*), denote H + PAR → G. Problem P3 has been found by the following steps*:*

- Step 1: Record the element in hypothesis and goal part
- Step 2: Check the goal G, if G is obtained then find a rule or a sample problem, which has hypothesis part is goal G, and the goal part is subset H then sent this rule (or sample problem) to step 7 and go go step 7.
- Step 3: Using heuristic rules to select a rule for producing new facts or new objects and go to step 2.
- Step 4: Using heuristic rules to select a sample problem for producing new facts or new objects and go to step 2.
- Step 5: Searching for any rules, which can be used to deduce new facts or new objects and go to step 2.
- Step 6: Giving conclusion: Solution not found, and stop program.
- Step 7: Giving R is the result from steps above. in this step we will consider R by argue the goal of R according to parameters. And then output result and stop program.

To find a sample problem in set of sample problem to resolve for problem on the COKB, in this *algorithm 3.5* we have used some heuristic rules to select a rule and sample problem below:

- Priority uses of rules for determine objects which have same object kind in set of goal.
- Priority uses of rules for determine new facts which have same kind of fact in set of goal.
- Priority use of rules for determining objects which have relationship with existing objects in set of goal.
- Priority uses of sample for determine new objects which have same object kind in set of goal.
- Priority uses of sample for determine new objects which have relationship with existing objects in set of goal.

V. Application

The structure of Intelligent Problem Solvers (IPS) system and processing to build them has been presented in [1]. It consists of following components:

- The knowledge base.
- The inference engine.
- The explanation component.
- The working memory.
- The knowledge manager.
- The interface.

Knowledge Bases contain the knowledge for solving some problems in a specific knowledge domain. It must be stored in the computer-readable form so that the inference engine can use it in the procedure of automated deductive reasoning to solve problems stated in general forms. They can contain concepts and objects, relations, operators and functions, facts and rules.

The Inference engine will use the knowledge stored in knowledge bases to solve problems, to search or to answer for the query. It is the "brain" that systems use to reason about the information in the knowledge base for the ultimate purpose of formulating new conclusions. It must identify problems and use suitable deductive strategies to find out right rules and facts for solving the problem. In an IPS, the inference engine also has to produce solutions as human reading, thinking, and writing.

The structure of IPS:

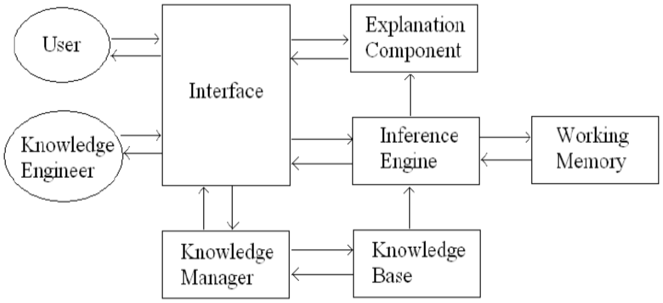


Fig. 1. Structure of a system

The main process for problem solving: From the user, a problem in a form that the user enter is input into the system, and the problem written by specification language is created; then it is translated so that the system receives the working problem in the form of the inference engine, and this is placed in the working memory. After analyzing the problem, the inference engine generates a possible solution for the problem by doing some automated reasoning strategies such as forward chaining reasoning method, backward chaining reasoning method, reasoning with heuristics. Next, the first solution is analyzed and from this the inference engine produces a good solution for the interface component. Based on the good solution found, the answer solution in human-readable form will be created for output to the user.

In this section, we will present some applications in Direct Current (DC) Electrical Circuits, Solid geometry and 2D-analytical geometry. The systems will be designed based on IPS, and knowledge-base component will be modeled and organized by COKB model, the inference engine will be designed based on reasoning algorithms in section III.B.

A. IPS in Direct Current Electrical Circuits

1) Design knowledge-base

Base on knowledge about DC Electrical Circuits has been mentioned in [19], this knowledge domain can be represented by COKB model as followed:

(C, Ops, Rules)

In this model, component H – hicherarchy and component R – relations have been lacked.

Set C of Concepts.

In knowledge domain about DC, Facts component in structure of Com-objects in C is an empty set. So this structure includes three components: **(Attrs, F, RulesObj)**

The set *C* consists of concepts such as "RESISTOR", "LAMB", "VariableResistor", "CIRCUIT", "CAPACITOR", "BATTERY".

<u>Eg. 4.1:</u> Structure of LAMB concept include:

Attrs = {□ R,V,I, P: values of resistance, potential difference, current flow, power of lamb
□ V_{max} , I_{max} , P_{max}: maximum values of potential difference, current flow, power of lamb
□ s: is a number performs three level of light, such as:
s = -1 : the light of lamb is low
s = 0 : the light of lamb is normal
s = 1 : the light of lamb is high}

$$F = \left\{ I = \frac{V}{R}, P = I^2 R, R = \frac{V_{max}^2}{P_{max}}, I_{max} = \frac{P_{max}}{V_{max}} \right\}$$

Rules={if (I < I_{max}) or (P < P_{max}) or (V < V_{max}) then s = -1
if (I = I_{max}) or (P = P_{max}) or (V = V_{max}) then s = 0
if (I > I_{max}) or (P > P_{max}) or (V > V_{max}) then s = 1
}

TABLE I. SET OPS OF OPERATORS BETWEEN CONCEPTS

Operator	Meaning	Arguments	Return	Properties
+	Circuits can be connected by series connection	CIRCUIT x CIRCUIT	CIRCUIT	commutative associative
//	Circuits can be connected by parallel connection	CIRCUIT x CIRCUIT	CIRCUIT	commutative associative
O	Capacitors can be connected by series connection	CAPACICATOR x CAPACICATOR	CAPACI-CATOR	commutative associative
Ξ	Capacitors can be connected by parallel connection	CAPACICATOR x CAPACICATOR	CAPACI-CATOR	commutative associative

In this knowledge domain, RESITOR concept can also be seen as a circuit with one resistor, so operator + and // can be applied on RESITOR concept, these properties and rules of two operators are unchanged.

Set Rules of rules.

Rules in this model are classified of two forms: deductive rules and equation rules.

+ <u>Some deductive rules in Rules-set:</u>

R1: *Rule of series circuits*
{D1, D2, AB: CIRCUIT, AB = D1 + D2}
→ {AB.V = D1.V + D2.V, AB.I = D1.I = D2.I,
AB.R = D1.R + D2.R}

R2: *Rule of parallel circuits*
{D1, D2, AB: CIRCUIT, AB = D1 // D2}
→ {AB.V = D1.V = D2.V, AB.I = D1.I + D2.I,
$\frac{1}{AB.R} = \frac{1}{D1.R} + \frac{1}{D2.R}$ }

R3: *Rule of series capacitors*
{Cap1, Cap2, Cap: CAPACICATOR, Cap = Cap1 o Cap2}
→ {Cap.Q = Cap1.Q = Cap2.Q, Cap.V = Cap1.V + Cap2.V,
$\frac{1}{Cap.C} = \frac{1}{Cap1.C} + \frac{1}{Cap2.C}$ }

R4: *Rule of parallel capacitors*
{Cap1, Cap2, Cap: CAPACICATOR, Cap = Cap1 Ξ Cap2}
→ {Cap.Q = Cap1.Q + Cap2.Q, Cap.V = Cap1.V = Cap2.V,
Cap.C = Cap1.C + Cap2.C}

+ <u>Some equation rules in Rules-set:</u>

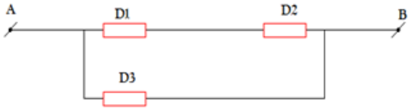
R5: D1 + D2 // D3 = (D1 + D2) // D3
R6: D1 // D2 + D3 = (D1 // D2) + D3
R7: C1 o C2 o C3 = (C1 o C2) o C3
R8: C1 Ξ C2 Ξ C3 = (C1 Ξ C2) Ξ C3

2) Model of problem and algorithm

Model of problem in this knowledge base about DC Electrical Circuits is defined as definition 3.1. Besides that, using algorithms 2.1, the inference engine has been built. This engine simulates the way of human thinking to find solution of practical problem.

3) Illustrating for solving problems

<u>Eg. 4.2:</u> _Three resistors are connected like the figure. Resistor D1 has value 30Ω, Resistor D3 has value 60Ω. The total current is 0.3 A, current though D3 is 0.2A .
What is the current though D1 ?



+ <u>Specification of Problem S:</u>
O:= {D1, D2, D3: RESISTOR ; AB: CIRCUIT}
F:= {AB = D1 + D2 // D3
D1.R=30, D3.R=60, D3.I=0.2, AB.I =0.3}
G:={D1.I}

+ <u>Solution of Program:</u>
STEP 1: D1 + D2 // D3 = (D1 + D2) // D3
by apply rule R5
STEP 2: Let [EF, CIRCUIT], EF = D1 + D2
AB = D1 + D2 // D3
D1 + D2 // D3 = (D1 + D2) // D3
➡AB = EF // D3

STEP 3: AB = EF // D3 → AB.I = EF.I + D3.I
by apply "Rule of Paralle Circuit" R2

STEP 4: EF = D1 + D2 →EF.I = D1.I
by apply "Rule of Series Circuit" R1

STEP 5: From { D3.I = 0.2, AB.I = 0.3, AB.I = EF.I + D3.I} → EF.I = 0.1
by "Solve equation"

STEP 6: From { EF.I = 0.1, EF.I = D1.I} → D1.I = 0.1

B. Intelligent problem solvers in Solid geometry

1) Design knowledge-based

Base on knowledge about Solid Geometry has been mentioned in [20], this knowledge base represented by COKB model consists of five components:

(C, H, R, Funcs, Rules)

The components of the model are listed as followed:

Set C of Concepts.

The set C consists of concepts of Com-objects. There is a variety of concepts in Solid geometry, including "Point", "Segment", "Line", "Angle", "Plane", "Triangle", "Equilateral Triangle", "Isosceles Triangle", "Right Triangle", "Isosceles Right Tri-angle", "Quadrilateral", "Trapezoid", "Right Trapezoid", "Parallelogram", "Rhombus", "Rectangle", "Square", "Triangular Pyramid", "Regular Triangular Pyramid", "Quadrilateral Pyramid" and "Square Pyramid".

Set H of hierarchical relations on the concepts C.

There are hierarchical relationships among the concepts in set C and they can be represented with using Hasse diagrams. For example, the following Hasse diagram demonstrates the hierarchy on the concepts of Quadrilaterals.

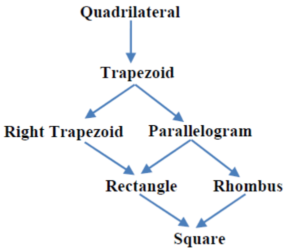


Fig. 2. The hierarchy on the concepts of Triangle

Set R of relations on Com-Objects.

Set R contains various kinds of relations among Com-Objects. The following are some examples of relations: Relation about Perpendicular between a Plane and a Plane, a Line and a Plane, a Line and a Line; Relation about Distinct between a Point and Point, a Line and a Line; Relation about Belong between a Point and a Segment, a Point and a Line, a Line and a Plane; Relation about Altitude between a Segment and a Triangle, a Segment and a Quadrilateral Pyramid.

Set Funcs of functions on Com-Objects.

The set Funcs consists of functions on Com-Objects, such as: Function *Line of Intersection* between two Planes returns a Line. Function *Point of Intersection* between two Lines, or between a Line and a Plane returns a Point. Function *Orthogonal Projection* of a Point onto a Line, or a Point onto a Plane returns a Point. Function *Orthogonal Projection* of a Line onto a Plane returns a Line. Function *Common Perpendicular* of two skew Lines returns a Line. Function *Distance* between a Point and a Line, a Point and a Plane, two Lines, a Line and a Plane, two Planes returns an integer number.

Set Rules of rules.

Rule 1: Three non-collinear points define a plane.
{A, B, C: Point; A, B, C are non-collonear}
⇨{Determine a plane (ABC)}

Rule 2: The intersection point of two diagonals of a parallelogram is the center of the parallelogram.
Parallelogram[ABCD], O = AC ∩ BD ⇨ O is center of Parallelogram[ABCD]

Rule 3: If a point belongs to any lateral edge of a triangular pyramid then the point is not on the base of the triangular pyramid.
SABC is a triangular pyramid; I ∈ SA ⇨ I ∉ plane(ABC)

Rule 4: If a point belongs to both two distinct lines, then the point is the point of intersection of both of them.
d, d1: Line; H:Point; H ∈ d; H ∈ d1 ⇨ H = d ∩ d1

Rule 5: A point belongs to a line and a Plane is the point of intersection of the line and the plane
A: Point; d: Line; P: Plane; A ∈ d; A ∈ P; d ⊄ P ⇨ A = d ∩ (P)

Rule 6: If the two distinct points belong to a plane, then the line made by those points lies on the plane.
A, B: Point; P: Plane; A ∈ P; B ∈ P; A, B are distinct ⇨ AB ⊂ (P)

Rule 7: If a point belongs to a line and this line is on a plane, then the point belongs to the plane.
K: Point; d: Line; P: Plane; K ∈ d; d ⊂ (P) ⇨ K ∈ (P)

Rule 8: If a point is not in a plane then any line containing the point does not lie on the plane. *d: Line; A: Point; P: Plane; A ∉ P; A ∈ d ⇨ d ⊄ P*

Rule 9: Negative rule.
B, B': Point; B, B' are two distinct points; B ∈ d; B = d ∩ (P) ⇨ B' ∉ (P)

Rule 10: Generated rule a point of intersection of two lines.
K, A, B, C: Point; A, B, C are non-collonear; K is a midpoint of AB; J is a midpoint of AC ⇒ Let M = BJ ∩ CK, M is a midpoint of CK; M is a midpoint of BJ

2) Model of problem and algorithm

Almost problem in Solid geometry can be modeled by: (O,F) → G, and we can apply algorithm 3.1 for design inference engineer for this application.

3) Illustrating for solving problems

Eg. 4.3: Given a parallelogram ABCD, a point S outside plane(ABCD), a point M between S and A, and a point N between S and B. Let O is intersection point of line AC and BD. Find the intersection of line SO and plane (CMN).

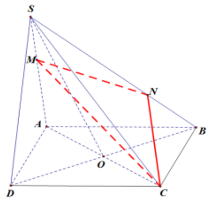


Fig. 3. Problem in eg. 4.3

O = { A, Point], [B, Point], [C, Point], [D, Point], [S, Point], [M, Point], [N, Point], [O, Point], Parallelogram[ABCD]

F = { S ∉ Plane[ABCD], ["Between", M, Segment[SA]], ["Between", N, Segment[SB]], O = PointOfIntersection(Line[AC], Line[BD]) }

G = { PointOfIntersection(Line[SO], Plane[CMN]) }

Solution found by the program: Note that, some steps of the following output solution is displayed in the human-readable form instead of using specification language. Therefore, the program helps users get easy to understand the solution.

STEP 1

Parallelogram[ABCD]; O = AC ∩ BD

⇨ O is center of Parallelogram[ABCD]

By rule *"The intersection point of two diagonals of a parallelogram is the center of the parallelogram"*

STEP 2

O is center of Parallelogram[ABCD]

⇨ O= MidPoint(A,C)

By rule *"The properties of the concept Square"*

STEP 3

O= MidPoint(A,C)

⇨ O ∈AC, O is between Segment[AC]

By rule *"The properties of function MidPoint"*

STEP 4

O ∈ AC; AC ⊂ Plane(SAC)

⇨ O ∈ plane(SAC)

By rule *"If a point belongs to a line and this line is on a plane, then the point belongs to the plane"*

STEP 5

M is between Segment[SA]

⇨ M ∈ SA

By rule *"The properties of relation Between"*

STEP 6

M ∈ SA; SA ⊂ Plane (SAC)

⇨ M ∈ Plane(SAC)

By rule *"If a point belongs to a line and this line is on a plane, then the point belongs to the plane"*

STEP 7

O ∈ Plane(SAC); S ∈ Plane(SAC)

⇨ OS ⊂ Plane(SAC)

By rule *"If the two distinct points belong to a plane, then the line made by those points lies on the plane"*

STEP 8

M ∈ Plane(SAC); C ∈ Plane(SAC)

⇨ CM ⊂ Plane(SAC)

By rule *"If the two distinct points belong to a plane, then the line made by those points lies on the plane"*

STEP 9

Triangle[SAC]; O is between Segment[AC]; M is between Segment[SA]

⇨ Let I= OS ∩ CM

By rule *"Generated rule a point of intersection of two lines"*

STEP 10

I = OS ∩ CM

⇨ I ∈ OS, I ∈ CM

By rule *"The properties of function Point of Intersection"*

STEP 11

I ∈ CM; CM ⊂ Plane(CMN)

⇨ I ∈ Plane(CMN)

By rule *"If a point belongs to a line and this line is on a plane, then the point belongs to the plane"*

STEP 12

C ∈ AC; C ∈ Plane(CMN)

⇨ C = AC ∩ Plane(CMN)

By rule *"A point belongs to a line and a Plane is the point of intersection of the line and the plane"*

STEP 13

O ∈ AC; C = AC ∩ Plane(CMN); O, C are distinct

⇨ O ∉ Plane(CMN)

By rule *"Negative rule"*

STEP 14

O ∈ SO; O ∉ Plane(CMN)

⇨ OS ⊄ Plane(CMN)

By rule *"If a point is not in a plane then any line containing the point does not lie on the plane"*

STEP 15

I ∈ OS; I ∈ Plane(CMN); OS ⊄ Plane(CMN)

⇨ I = OS ∩ Plane(CMN)

By rule *"A point belongs to a line and a Plane is the point of intersection of the line and the plane"*

C. IPS in 2D- analytical geometry

1) Design knowledge-base

Base on knowledge about 2D-Analytic Geometry has been mentioned in [21], this knowledge base represented by COKB model:

(C, H, R, Ops, Funcs, Rules)

The components of the model are listed as follows:

Set C of Concepts.

The set C consists of concepts of Com-objects. There is a variety of concepts in Euclid geometry, including "Point", "Segment", "Line", "Angle", "Triangle", "Equilateral Triangle", "Isosceles Triangle", "Right Triangle", "Isosceles Right Tri-angle", "Quadrilateral", "Trapezoid", "Right Trapezoid", "Parallelogram", "Rhombus", "Rectangle", "Square", "Circle", "Elip", "Hyperbol" and "Parabol".

Set H of hierarchical relations on the concepts

There are hierarchical relationships among the concepts in set C and they can be represented with using Hasse diagrams.

For example, the following Hasse diagram demonstrates the hierarchy on the concepts of Triangle.

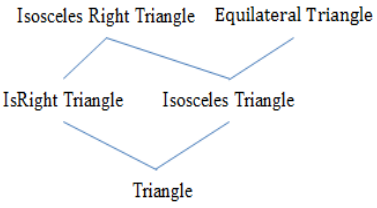


Fig. 4. The hierarchy on the concepts of Triangle

Set R of relations on Com-Objects

Between C-Objects, there are various kinds of relations. We have some relations: Relations of basic level: are relations between basic objects and objects of first level. Relations of first level: are relations between basic objects, objects of first level and objects of second level, or relations between objects of higher levels. Example: Relation Belong between a Point and a Segment, a Point and a Line, a Line and a circle, Segment and Circle, Line and circle…

Ops–set of relations on C-Objects

In knowledge about analytical geometry, we have operators on vectors such as: Operator *create vector* ($\overrightarrow{\quad}$) from two points it will return a vector. Operator *addition* (+), *subtraction* (-) of vectors it will return a vector. Operator *multiplication* (*) between two vectors, or between a vector with a coefficient.

Set Funcs of functions on Com-Objects

The set Funcs consists of functions on Com-Objects, such as: The set Funcs consists of functions on Com-Objects, such as: Function *distance* between two points it will returns a REAL number. Function *distance* between point and line. Function *symmetry* of a point through a point, or point through line. Function *mid point* of a segment. Function *mid angle* two lines, or two segments, or tow vectors. Function *projection* of a point into a line …etc.

Set Rules of rules

Almost properties, clauses, theorems in geometry can be represented by rules on facts relating to C-Objects. Followings are some particular rules:

{v: vector , u: vector, n: vector, u // v, u ⊥ n} ⇒ { v ⊥ n}

{u, n: vector, u ⊥ n} ⇒ { u * n = 0},

{AH: segment, ABC:Triangle, AH is altitude of triangle ABC } ⇒ {$\frac{1}{AH^2} = \frac{1}{AB^2} + \frac{1}{AC^2}$},

{ABCD: Parallelogram} ⇒ {$\overrightarrow{GB} + \overrightarrow{GC} = \overrightarrow{GD}$},

{AD: Segment, ABC: Triangle} ⇒ {D is midpoint of BC, BD = DC}, etc

2) Model of problem and algorithms

Model of problems in knowledge domain about 2D-analytic geometry can be represented by models in definition 3.1- 3.3. The inference engine has been also designed base on algorithms correlatively.

3) Illustrating for solving problems

<u>Eg. 4.4</u>**:** Giving two point B(2;1), C(6;1)

a. Determine A(x,y), with condition, (x>0, y>0) and ABC is equilateral triangle.

b. Determine A' symmetry with A through C.

c. Determine D, with condition $\overrightarrow{AD} - 3\overrightarrow{BD} + 4\overrightarrow{CD} = \overrightarrow{0}$, d. Find M to ABCM is parallelogram. Determine the center of ABCM.

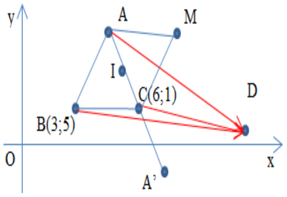


Fig. 5. Problem in Eg 4.4

This problem can be modeled as follows:

O = { [A,POINT],[B,POINT],[C,POINT],
[D,POINT],[M,POINT],[A',POINT],
[I,POINT]}
F = { B=[2,1], C=[6,1]}

$O_{condition}$ = { EQUILATERAL_TRIANGLE[A,B,C],
PARALLELOGRAM[A,B,C,M]
}
$F_{condition}$ =
{VECTOR[AD]-3VECTOR[BD]+4VECTOR[CD]=0,
A'=SYMMETRY(A,C), A.x>0, A.y>0,
["CENTER",I, PARALLELOGRAM[A,B,C,M]]}
G = {Determine: A, A', D, M}

+ <u>Solution of program:</u>

STEP 1
{[EQUILATERALTRIANGLE[A,B,C],
EQUILATERALTRIANGLE]}
→{DISTANCE(A,B)=DISTANCE(A,C),
DISTANCE(A,C)=DISTANCE(B,C),
DISTANCE(A,B)=DISTANCE(B,C)}
By: rule "properties of EQUILATERAL TRIANGLE"

STEP 2
{A=[A.x, A.y], B=[2,1], DISTANCE(A,B)}
→{DISTANCE(A,B)= $\sqrt{(A.x-2)^2+(A.y-1)^2}$}
By: function "*solving of function distance*" of two points

STEP 3
{ A=[A.x, A.y], C=[2,1], DISTANCE(A,C)}
→{DISTANCE(A,C)= $\sqrt{(A.x-6)^2+(A.y-1)^2}$}
By: function "*solving of function distance*" of two points

STEP 4
{B=[2,1], C=[6;1], DISTANCE(B,C)}
→{DISTANCE(B,C)=4}
By: function "*solving of funtion distance*" of two points

STEP 5
{DISTANCE(B,C)=4,
DISTANCE(A,B)= $\sqrt{(A.x-2)^2+(A.y-1)^2}$,
DISTANCE(A,C)= $\sqrt{(A.x-6)^2+(A.y-1)^2}$,
DISTANCE(A,B)=DISTANCE(A,C),
DISTANCE(A,C)=DISTANCE(B,C),
DISTANCE(A,B)=DISTANCE(B,C)}
→{$(A.x-2)^2+(A.y-1)^2=4$, $(A.x-6)^2+(A.y-1)^2 = 4$}

By: Equation “ *auto create equations*” between two equations

STEP 6

$\{(A.x-2)^2+(A.y-1)^2=4,\ (A.x-6)^2+(A.y-1)^2 = 4\}$
→ {*A*=[4,1+2√3] }
By: Equations “*solve system of equations*” between two equations

STEP 7

{A' = SYMMETRY(A,C), C = [6 , 1], A=[4,1+2√3]}
→{A'=[8,1-2√3]}
By: function “*solving of function symmetry*” of two points

STEP 8

{A = [4 , 1+2√3], D = [D.x, D.y]}
→{VECTOR[AD]=[D.x-4, D.y-(1+2√3)]}
By: operators “*create vector*” of two points

STEP 9

{B = [2 , 1], D = [D.x, D.y]}
→{VECTOR[BD]=[D.x-2, D.y-1]}
By: operators “*create vector*” of two points

STEP 10

{C = [6 , 1], D = [D.x, D.y]}
→{VECTOR[CD]=[D.x-6, D.y-1]}
By: operators “*create vector*” of two points

STEP 11

{VECTOR[AD]-3VECTOR[BD]+4VECTOR[CD] = VECTOR[0],
VECTOR[AD]=[D.x-4, D.y-(1+2√3)],
VECTOR[CD]=[D.x-6, D.y-1],
VECTOR[BD]=[D.x-2, D.y-1]}
→{(2D.x-22=0, 2D.y-2- 2√3=0}
By: expression of vector *“compute expression of vector”*

STEP 13

{(2D.x-22=0, 2D.y-2- 2√3=0}
→ {D=[11,1-√3]}
By: System of Equations “*solve system of equations*” between two equations

STEP 14

{[PARALLELOGRAM[A,B,C,M], PARALLELOGRAM]}
→ {VECTOR[AM] = VECTOR[BC]}
By: rules “*properties of parallelogram*”

STEP 15

{A = [4,1+2√(3)], M =[M.x, M.y]}
→ {VECTOR[A,M] = [M.x -4, M.y -1+2√3]}
By: operator “*create vector*” from two points

STEP 16

{B = [4,1], C =[6, 1]}
→ {VECTOR[B,C] = [2,0]}
By: operator “*create vector*” from two points

STEP 17

{VECTOR[AM] = VECTOR[BC]
VECTOR[A,M] = $[M.x\ -4,\ M.y\ -1+2\sqrt{3}]$
VECTOR[B,C] = [2,0]}
→ {M = [8,1+2√3]}
By: sample problem “two *equal vectors*”

STEP 18

{["CENTER",I,PARALLELOGRAM[A,B,C,M]]}
→ {I=MIDPOINT(A,C)}
By: rule “*properties of parallelogram*”

STEP 19

{I=TRUNGDIEM(A,C),
A = [4,1+2√(3)], C = [6 , 1]}
→{I(5;1+√3)}
By: function “*solving of function midpoint*” of two points.

Eg. 4.5: Giving circle (C) with center I and radius 4(cm) , equation of (C): $(x-2)^2+(y-3)^2=4$, give point: A(5;2), B(0,m), let determine m to AB intersect with O.

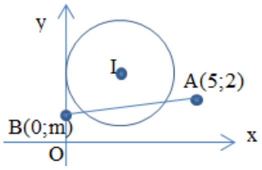


Fig. 6. Problem in Eg.4.5

This problem can be modeled as follows:

O = {[A, POINT], [B, POINT], [I, POINT],
[CIRCLE[I,2],CIRCLE], [LINE[A,B],LINE]}
F = {A=[5;2], B=[0;m]),
$C.expr=[(x-2)^2+(y-3)^2=4]\}$
PAR = {m}
G = {
Determine:[m, ["INTERSECTION",LINE[AB],CIRCLE[I,2]]]
}

+ Solution of program:

STEP 1

{ A=[5,2], B=[0,m]}
→{LINE[AB].expr=[(m-2)x+5y-5m=0]}
By: sample problem “*create equation of line*” from two point

STEP 2

{["INTERSECTION",LINE[AB],CIRCLE[C,2]]}
→{DISTANCE(C.I,LINE[AB]) < 2}
By: rule “*relative position of line and circle*” between a line and a circle

STEP 3

$\{C.expr=[(x-2)^2+(y-3)^2=4]\}$
→{C.I=[2;3]}
By: rule “*properties of circle”*

STEP 4

{DISTANCE(C.I,LINE[AB]),
C. I=[2;3],
LINE[AB].expr=[(m-2)x+5y-5m=0]}
→{DISTANCE(C.I, LINE[A,B])= $\frac{(-3m+11)}{\sqrt{(m-2)^2+25}}$}
By: function “*solving of function distance*” from a point to a line

STEP 5

{DISTANCE(C.I,LINE[AB])< 2,
DISTANCE(C.I, LINE[A,B])= $\frac{(-2m+8)}{\sqrt{(m-2)^2+16}}$}
→ {m>5-2√6}
By: solve of inequality “*solve of inequality according to parmeters*”

STEP 6

Conclude: with {m>5-2√6} then
["INTERSECTION",LINE[AB],CIRCLE[I,2]]]

VI. Conclusion

In previous studies, knowledge about operators and functions have been presented, but they are still discrete and do not research in the model that have many components with concepts and relations between them. In this paper, operators and functions components have been researched in the context of knowledge structure as ontology. Results of this research are: representation, specification, classified kinds of facts relative to operators and functions and processing about unification of facts, model of problems, reasoning methods. With these technologies, scientific basis for designing knowledge base of IPS has been solved.

COKB model with researching about Ops and Funcs component can be used to design and implement intelligent problem solvers. It provides a natural way for representing knowledge domains in practice. By integration of ontology engineering, object-oriented modeling and symbolic computation programming it provides a highly intuitive representation for knowledge. These are the bases for designing the knowledge base of the system. The methods of modeling problems and algorithms for automated problem solving represent a normal way of human thinking. Therefore, systems not only give human readable solutions of problems but also present solutions as the way people write them. COKB and reasoning method on knowledge about Functions and Operators have been applied to construct IPS in education, as: Direct Current Electrical Circuits, Solid Geometry, 2D-Analytic Geometry. The solution given by programs is step-by-step, natural, precise and has reasoning like human.

Our future works are to develop the models and algorithms:

First, different types of COKB model will be researched: reducing of COKB, extension of COKB and integration many types of COKB in real applications.

Second, the rules component of COKB model should have classified by many criteria, such as: deductive rules, equivalent rules, equation rules and reasoning methods base on rules.

Third, researching about relations combines to operators and functions components. In which, it is usefully to research about the coordinate of knowledge about relations, operators and function in knowledge domain that have to using combination these kinds of knowledge.

ACKNOWLEDGMENT

This research in funded by Vietnam National University HoChiMinh City (VNU-HCM) under grant number C2014-26-02.

REFERENCES

[1] Nhon V. Do. *Intelligent Problem Solvers in Education: Design Method and Applications, Intelligent Systems*, Prof. Vladimir M. Koleshko (Ed.), ISBN: 978-953-51-0054-6, InTech (2012).

[2] Michel Chein & Marie-Laure Mugnier, "Graph-based Knowledge representation: Computational foundations of Conceptual Graphs", Springer-Verlag London Limited, 2009

[3] Chitta Baral, *Knowledge Representation, Reasoning and Declarative Problem Solving*, Cambridge University Press, 2003.

[4] B. Plotkin, T. Plotkin, *An algebraic approach to knowledge base models informational equivalence*, Acta Appl. Math. Vol. **89**, no. 1-3, 109–134 (2005).

[5] Marina Knyazhansky, *Knowledge Bases over algebraic models: Some notes about information equivalence*, Intenational Journal of Knowledge Management, Vol. 8, Issue 1, ISSN: 1548-0666. (2012)

[6] E. Roanes-Lozano, L. M. Laita, A. Hernando, E. Roanes-Macias, *An algebraic approach to rule based expert systems*, Revista de la Real Academia de Ciencias Exactas, Fisicas y Naturales. Serie A. Matematicas (RACSAM), March 2010, Vol. 104, Issue 1, pp. 19-40, Springer (2010)

[7] P. Cordero, G. Gutiérrez, J. Martínez, I.P. de Guzmán , *A New Algebraic Tool for Automatic Theorem Provers*, Annals of Mathematics and Artificial Intelligence, Vol. 42, Issue 4, pp. 369-398 (2004)

[8] Riichiro Mizoguchi, *Advanced course of Ontological Engineering*, New Generation Computing, Vol. 22, Issue 2 (2004), 193-220, Ohmsha Ltd. and Springer-Verlag, 2004.

[9] John F. Sowa, "Knowledge Representation: Logical, Philosophical and Computational Foundations", Brooks/Cole, 2000

[10] Bo Liu, Jianqiang Li, Yu Zhao, "A Query-Specific Reasoning method for Inconsistent and Uncertain Ontology", *2011 International MultiConference of Engineers and Computer Scientists* (IMECS 2011), ISBN: 978-988-18210-3-4, pp. 53-58, Hongkong, March 2011

[11] Nhon V. Do, "*Ontology COKB for designing knowledge-based systems*", Proceeding of 13th International Conference on Intelligent Software Methodologies, Tools, and Techniques (SOMET 2014), pp. 354-373, *New Trends in Software Methodologies, Tools and Techniquies*, Hamido Fujita et al. (eds.), IOS press. (2014)

[12] Nhon V. Do, Hien D. Nguyen, Thanh T. Mai, *Designing an Intelligent Problems Solving System based on Knowledge about Sample Problems*, Proceeding of 5th Asian conference on Intelligent Information and Database Systems (ACIIDS 2013), Kuala Lumpur, Malaysia, March 2013, LNAI 7802, pp. 465-475, Springer

[13] Thomas Bitter, Maureen Donnelly, Computational ontologies of parthood, component hood, and containment, *Proceeding of International Joint Conferences on Artificail Intelligent 2005* (IJCAI 2005), Edinburgh, Scotland, UK.

[14] Jakub M. Tomczak, Jerzy Swiatek, *Knowledge acquisition method for Realational Knowledge Representaion* (2009)

[15] Nhon V. Do, Hien D. Nguyen, *A Knowledge Model about Relations and Application*. Proceeding of 6th IEEE International Conference on New Trends in Information Science, Service Science and Data Mining (ISSDM 2012), pp. 707-710, Taipei. 2012

[16] Lakemeyer, G. & Nebel, B., 1994, *"Foundations of Knowledge representation and Reasoning"*. Berlin Heidelberg: Springer-Verlag.

[17] Nhon Do, Hien Nguyen, *A reasoning method on Computation Network and Its applications*, 2011 International MultiConference of Engineers and Computer Scientists, IMECS 2011, ISBN: 978-988-18210-3-4, pp. 137-141, Hongkong , March 2011.

[18] Van Nhon Do, Diem Nguyen, *Intelligent Problem Solving about Functional component of COKB model and Application*. D. Camacho et al. (eds.), *New Trends in Computational Collective Intelligence*, pp. 27-37, Studies in Computational Intelligence 572, Springer (2015)

[19] Vietnam Ministry of Education and Training, *Textbook and workbook of Direct Current Electrical Circuits in middle school*, Publisher of Education (2011).

[20] Vietnam Ministry of Education and Training, *Textbook and workbook of Solid Geometry in high school*, Publisher of Education (2012).

[21] Vietnam Ministry of Education and Training, *Textbook and workbook of 2D-Analytic Geometry in high school*, Publisher of Education (2012).

[22] UMS - Math: http://www.umsolver.com/cgi-bin/club.pl?language=en

[23] Mathway: https://mathway.com/