CUBIC INTERPRETATION OF RESOLUTION PRINCIPLE FOR PROPOSITIONAL CALCULAS

Semerenko V.P.
State Technical University, Vinnitsa
95, Khmelnitskoe shosse, STU, Vinnitsa, 286021, Ukraine
phone: (380 432) 44-03-79

Abstract

Interpretation of resolution principle which is able to realize the resolution procedure for any formula of propositional calculas during finite time and with the possibility of parallel computing are considered.

***Keywords**: artificial intelligence, propositional calculas, resolution principle, boolean algebra, parallel processing.*

Introduction

Software and hardware development in the field of artificial intelligence requires a complex approach in solving all the arising problems, such as the choice of knowledge representation and processing method, proof procedure formalization, the designing of the device and their production technology.

For each of the above mentioned problems there are many known ways of solving them, however, there is a need to find among them an optimal combinational which can lead to better and new development.

Realization of predicate calculas without variables, and resolution principle of Robinson which are able to formalize the inference method oriented to perspective hardware.

Logic inference in the algebra of cubic functions

In the propositional calculas the inference rule from finite chain of formulas (antecendents) $A_1, A_2, \dots, A_n$ of same formula (consequent) can be put down by means of implication operator:

$$(A_1, A_2, \dots, A_n) \rightarrow C \ , \qquad (1)$$

which is logically equivalent to verification of unsatisfiability formula:

$$A_1 \wedge A_2 \wedge \dots \wedge A_n \wedge \overline{C} \ , \qquad (2)$$

or verification of satisfiability formula:

$$\overline{A_1} \vee \overline{A_2} \vee \ldots \vee \overline{A}_n \vee C \ . \qquad (3)$$

Resolution principle uses formula (2) which is inverse to formula (1), however, for the process of proving both the formula (2) and formula (3) can be used if the necessary practical result must be obtained as quick as possible.

The algorithm of unsatisfiability verification of the formula (2) is undeterminable and work without time limitations in case formula (2) is not valid. Different modification of resolution principle use additional information for the acceleration of proof search, though they are complicated for the computer realization.

Let's consider the modification variant of resolution principle which is used to obtain an effective hardware realization.

The first problem which it was necessary to solve in connection with previously mentioned, was a convenient form of presentation input data.

Having this in mind let's analise the propositional calculas realization with the help of boolean algebra of cubic functions [4]. The suggested algebra is isomorphous to the traditional algebra of logic functions.Conjunction, disjunction and negation operations in the algebra of logical functions correspond to the operations of intersection, union and difference of cubs in the algebra of cubic functions and to normal forms correspond cubic coverings. Cubic D-covering (cubic R-covering) corresponds to a minimal dusjunctive normal form of same logical function f (inverse function $\bar{f}$), which is represented in the cubic form, i.e. by means the alphabet { 0,1,x }.Similarly, cubic K-covering (cubic Q-covering) corresponds to minimal conjunctive normal form of same
logical function f (inverse functions $\bar{f}$).

Similarly by three classes of propositional calculas formulas there are three classes of functions in boolean algebras: valid, unsatisfiable and satisfiable functions. With the help of D-covering (Q-covering) only valid and satisfiable functions can be represented where as only unsatisfiable and satisfiable functions can be represented with the help of R-covering (K-covering).

Now, from the point of view of algebra of cubic functions the proving according to the (2) and (3) formulas can be considered.

Formula (2) is usually represented as a list of (n+1) disjuncts $A_1, A_2, \ldots, A_n, \overline{C}$ between each pair of them resolvents are formed. Then disjuncts $A_1, A_2, \ldots, A_n, \overline{C}$ can be placed in accordance with coverings $D_1, D_2, \ldots, D_n, R_{n+1}$ (if D_{n+1} - covering for disjunct C).Coordinate number m in coverings $D_1, D_2, \ldots, D_n, R_{n+1}$ is equal to

the number m of different symbols in disjuncts $A_1, A_2, \dots, A_n, \overline{C}$, but the number of cubs in the covering $D_i(R_{n+1})$ is equal to the number of symbols in the disjunct $A_i(\overline{C})$. The value 1 in the h-th coordinate of the covering $D_i(R_{n+1})$ corresponds to the presence of the h-th symbol in the disjunct $A_i(\overline{C})$,and the value 0 - to the presence of the h-th symbol with the inversion in the disjunct $A_i(\overline{C})$ ($h=1\div m$). For example, the production rule

$$(a\vee\overline{b},\ \overline{a}\vee c,\ b\vee\overline{c}\vee d)\to d \qquad (4)$$

will be presented as the following coverings:

$$D_1=\begin{matrix} a & b & c & d \end{matrix}\begin{bmatrix} 1 & x & x & x \\ x & 0 & x & x \end{bmatrix};\ D_2=\begin{matrix} a & b & c & d \end{matrix}\begin{bmatrix} 0 & x & x & x \\ x & x & 1 & x \end{bmatrix};\ D_3=\begin{matrix} a & b & c & d \end{matrix}\begin{bmatrix} x & 1 & x & x \\ x & x & 0 & x \\ x & x & x & 1 \end{bmatrix};\ R_4=\begin{matrix} a & b & c & d \end{matrix}\begin{bmatrix} x & x & x & 0 \end{bmatrix} \qquad (5)$$

In the algebra of cubic function analogous procedure of define resolvent disjunct between disjuncts A_i and A_j is the operation of cubic intersection (¿-operation) of corresponding coverings:

$$D_{i,j}=D_i\cap D_j,\ i,j=1\div n, i\neq j\ . \qquad (6)$$

Formal definition of cubic intersection operation is presented in [5]. For example, for two pairs of cubs ($0\ x\ 1\ x\ 1$), ($x\ 1\ x\ x\ 1$) and ($x\ 1\ 0\ x\ x$), ($1\ 0\ x\ 0\ 1$) the results of their intersection will be following:

$$\begin{array}{l} 0\ \ x\ \ 1\ \ x\ \ 1 \\ 0\ \ 1\ \ 1\ \ x\ \ 1 \\ \text{----------} \\ 0\ \ 1\ \ 1\ \ x\ \ 1 \end{array} \qquad \begin{array}{l} x\ \ 1\ \ 0\ \ x\ \ x \\ 1\ \ 0\ \ x\ \ 0\ \ 1 \\ \text{----------} \\ \varnothing\ , \quad \text{i.e empty.} \end{array}$$

The covering $D_{i,j}$ obtained as a result of operation (6) which will be called resolvent may either contain new cubs or be empty.

The class of formulas (1) and (2) can be defined by means of following theorem.

THEOREM 1: Let the following coverings correspond to the formulas $A_1, A_2, \dots, A_n, \overline{C}$:

$$D_1, D_2, \dots, D_n, R_{n+1}\ , \qquad (7)$$

and D' is the result of coverings (7) intersections:

$$D'=D_1\cap D_2\cap\ ,\dots,\cap D_n\cap R_{n+1} \qquad (8)$$

Then formula (1) is valid and the formula (2) is unsatisfiable if:

$$D'=\varnothing\ ,$$

and both formulas (1) and (2) are satisfiable if:

$$D'\neq\varnothing\ .$$

In the same way the class of formulas (1) and (3) can be defined by means of the following theorem.

THEOREM 2: Let the following coverings correspond to the for-

mulas $\overline{A_1}, \overline{A_2}, \dots, \overline{A_n}, C$:

$$R_1, R_2, \dots, R_n, D_{n+1}\ , \qquad (9)$$

and $D^{''}$ is the result of coverings (9) intersections:

$$D^{''} = R_1 \cap R_2 \cap , \dots , \cap R_n \cap D_{n+1} \qquad (10)$$

Then formula (1) is valid and the formula (3) is valid if:

$$D^{''} = I\ ,$$

and both formulas (1) and (3) are satisfiable if:

$D^{''} \neq I$, where $I = [x \quad x \quad \dots \quad x]$ - m -cub.

The verification of the formula of propositional calculas for satisfiability, as in [4],will be called resolution procedure (RP).

Traditional RP based on the resolution principle can be endless as it does not have a simple indicator of terminate for the satisfiable formulas. Theorems 1 and 2 given above allow us to work out RP which is effective and can be terminate during finite number of steps for any class of formulas. Without any loss of generability let's confine ourselves to finding of RP only on the basis of theorem 1.

As it is shown in [3], the RP can be substituted to a classic problem of search. In fact, RP can be considered as a process of search at least of one cub covering $D^{'}$ or $D^{''}$. The presence of such cub will be evidence that formulas (1),(2) and (3) are satisfiable.

Let's consider two RP: serial RP (on the basis of the serial search problem) and binary RP (on the basis of the binary search problem).

Serial RP is realized by means of iterative definitions of resolvent covering in the following way:

$$D_{1,2} = D_1 \cap D_2\ ,\quad D_{1,2,3} = D_{1,2} \cap D_3\ ,\ \dots\ ,\quad D^{'} = D_{1,2,\dots,n} \cap R_{n+1}\ .$$

The procedure is ended after an empty resolvent covering $D_{i,j}$ has been obtained during one of the iterations, because then:

$$D^{'} = D_{i,j} = \emptyset\ .$$

For the verification of any valid or unsatisfiable formula set by coverings (7) from 1 to n iterations may be required and for the verification of a satisfiable formula always n iterations are necessary.

So in the worst case the serial RP is substituted to exhaustive search, however such RP will always be completable with concrete result in finite number of steps.

The fastest procedure of propositional calculas formula class defining is a binary RP. It can be graphically drawn as a binary search tree $G(V,E)$, consisting of the set of nodes V and the set of edges E (Fig.1).

Every leaf node $v_1^1, \dots, v_{n+1}^1$, ($v_i^1 \in V, i=1 \div n+1$)of the tree $G(V,E)$ is maped to one of the coverings $D_1, D_2, \dots, D_n, R_{n+1}$, and the root v_0 of that tree is maped to covering D'. The internal node $v_{i,j}^2$ of the second level, connected with the nodes v_i^1 and v_j^1 by edges is maped to the resolvent covering $D_{i,j}$ (6). Similarly every internal node of the k-th level is in the same way is maped to the resolvent covering, obtained as a result of the intersection operation of two resolvent coverings from ($k-1$)-th level.

Such a tree is maped to the recursive algorithm, which builds binary search tree in the opposite direction that is from the leaf node to the root. From the theoretic point of view, the binary RP requires the same quantity of iterations, as the serial, but the use of cubic presentation properties may give an essential practical gain in time.

PROPERTY 1: If there are only 'x' and '1' or 'x' and '0' values in h-th coordinate, then formulas (1) and (2) are satisfiable ($h=1 \div m$).

PROPERTY 2: If there are values 'x', '0' and '1' in all coordinates of all coverings then formula (1) may be satisfiable or valid and formula (2) may be satisfiable or unsatisfiable.

PROPERTY 3: Formula (1) will be unsatisfiable and formula (2) will be valid, if resolvent covering is empty and corresponding to any internal node of tree G(V,E).

Properties like properties 1-3, can be formalized in regard to coverings (9), in order to distinguish satisfiable and valid formulas.

PROPERTY 4: The operation of cubic intersection for cub $a=(a_1 a_2 \dots a_m)$ and cub $b=(b_1 b_2 \dots b_m)$ consists of m elementary cubs components intersection operations $a_h \cap b_h, \ (h=1 \div m)$, the results of which are independent of each other.

The fourth property defines the most important advantage of cubic data presentation, allowing to organize parallel data processing.

As a result the next algorithm for the definition of the class of formulas (1) and (2) on the basis of theorem 1 can be suggested. The initial data for such algorithm is the set M which consists of the coverings (7).

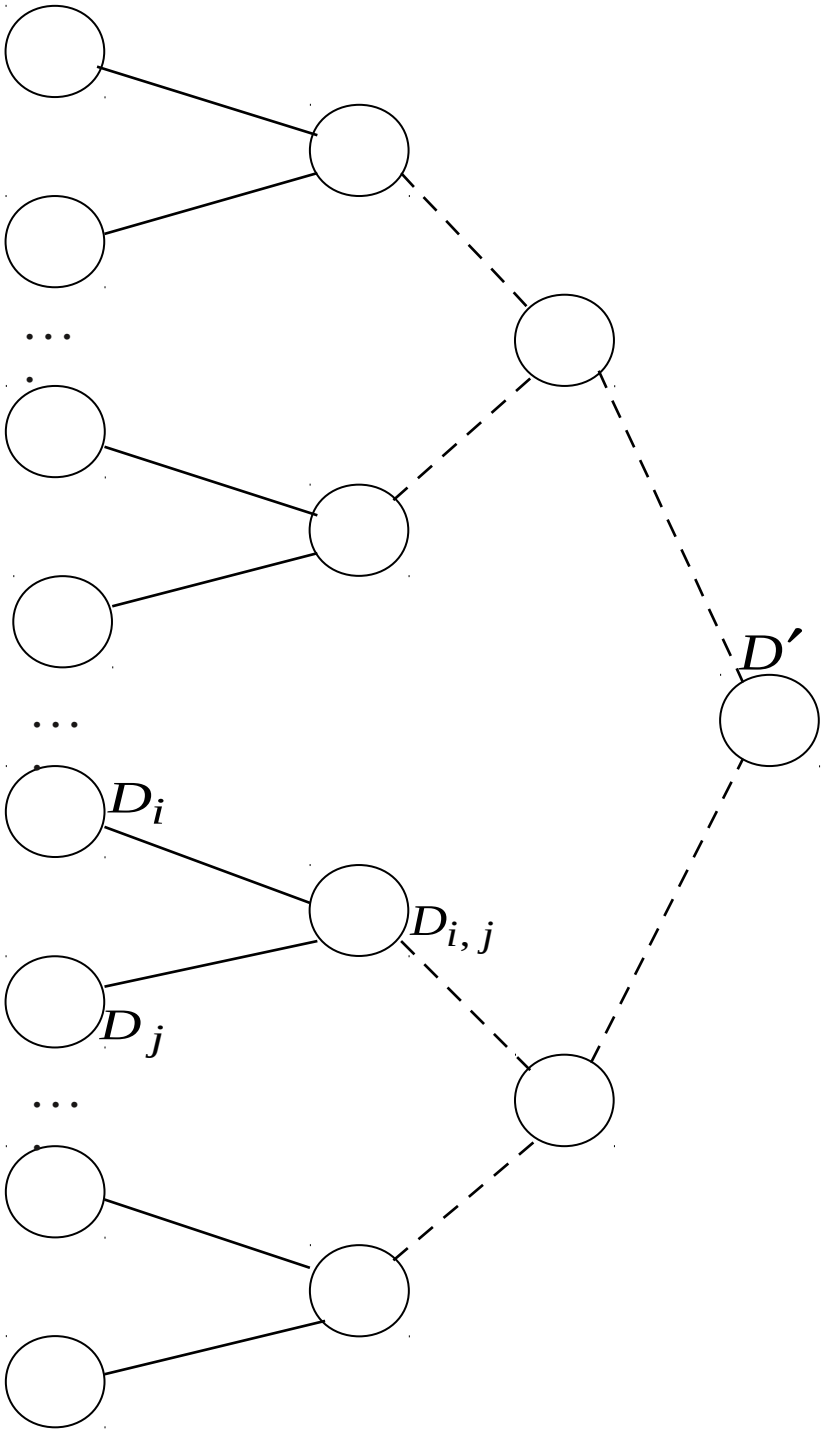

Fig. 1. Tree

Algorithm of resolution procedure

1. Create two empty sets M_1 and M_2. Let h =1.

2. Select from set M any covering $D^{(h)}$ which contains ones (zeros) in h-th coordinate. If such coverings do not exist, then go to 3, otherwise go to 4.

3. Let $h=h+1$. If $h \le m$ then return 2, otherwise go to 6.

4. Select from set M any covering $D^{(\bar{h})}$ which contains zeros (ones) in h-th coordinate. If such coverings do not exist then return to 3, otherwise go to 5.

5. Transfer covering $D^{(h)}$ from set M to set M_1 and covering $D^{(\bar{h})}$ from set M to set M_2. Return to 3.

6. If set M is empty, then go to 8, otherwise go to 7.

7. Transfer remained coverings from set M to set M_1 and M_2 and distribute them between M_1 and M_2 so that $|M_1|=|M_2|$. (It is admited that sets M_1 and M_2 may be crossable).

8. Let t =1 and $s=|M_1|=|M_2|$.

9. Select t-th pair coverings D_i, D_j from sets M_1 and M_2 and define resolvent covering $D_{i,j}$:

$$D_{i,j}=D_i \cap D_j,\ D_i \in M_1, D_j \in M_2 .$$

10. If $D_{i,j}=\emptyset$, then go to 14, otherwise put covering $D_{i,j}$ into set M and go to 11.

11. Let $t=t+1$. If $t>s$, then go to 12, otherwise go to 9.

12. If set M contains one covering then go to 13, otherwise go to 1.

13. Formula 1 is satisfiable. Go to 15.

14. Formula 1 is valid.

15. End.

This algorithm builds the binary search tree during w ($w=\log_2(n+1)$) steps, whereas at k-th step set M contains coverings which correspond to the nodes k-th level of tree ($k=1 \div w$). Algorithm stops its work when empty resolvent covering is obtained for the first time (if formula (1) is valid) or when it builds full tree (if formula (1) is satisfiable).

During the first partitioning of the set M into two equal sets M_1 and M_2 not more than one unit or one zero can be marked on each coordinate of input coverings. Several ones are zeros can appear on one coordinate of resolvent coverings, whereas ones and zeros can appear simultaneously. In the last case, when items 2 and 4 of the algorithm are fulfiled, the choice of necessary

covering on h-th coordinate is performed according to the majority of ones or zeros.

Let's consider the implementation of the algorithm for the definition of the class formula, which are set by covering (5). At the beginning set M contains input coverings: $M=\{D_1, D_2, D_3, R_4\}$. According to the first coordinate (coordinate 'a') following coverings will get into the created sets M_1 and M_2:

$$M_1=\{D_1\}, \quad M_2=\{D_2\}.$$

Further the remained coverings will be distributed between two sets M_1 and M_2 according to the last coordinate (coordinate 'd'):

$$M_1=\{D_1,D_3\}, \quad M_2=\{D_2,R_4\}.$$

After that we find two resolvent coverings:

$$D_{1,2}=D_1\cap D_2=\begin{matrix} a & b & c & d \end{matrix}\begin{bmatrix} 1 & x & 1 & x \\ 0 & 0 & x & x \\ x & 0 & 1 & x \end{bmatrix}, \qquad D_{3,4}=D_3\cap R_4=\begin{matrix} a & b & c & d \end{matrix}\begin{bmatrix} x & 1 & x & 0 \\ x & x & 0 & 0 \end{bmatrix}.$$

The obtained resolvent coverings get into the opposite sets:

$$M=\{D_{1,2}, D_{3,4}\}, \quad M_1=\{D_{1,2}\}, \quad M_2=\{D_{3,4}\}.$$

At last we find the new resolvent covering:

$$D_{1,2}\cap D_{3,4}=\begin{matrix} a & b & c & d \end{matrix}\begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

As soon as one not empty resolvent covering is obtained, consequently the formula, which is discussed here, is satisfiable.

The folloving ways of accelerating RP execution can be proposed.

1.By the inspection of the original coverings (7) or (9) the situation is looked for when there are only 'x' and '1' values or 'x' and '0' values on one of the coordinates. Then according to the property 1 such formula will be satisfiable and the application of the algorithm will be unnecessary.

2.Three kinds of parallelism are used:

- parallel fulfilment of elementary intersection operations between separate cub's components according to the property 4;
- parallel fulfilment of intersection operations between pairs of cubs, when resolvent covering is found;
- parallel defining of resolvent coverings on each level of binary search tree.

Summary

The use of cubical data representation allows to solve different special problems of artificial intelligence from the same approach.

Firstly,the propositional calculas obtains a very convenient presentation from computer realization.In the cubs language the logic of predicates with variables can be also easily expressed, though it is the topic for another paper.

Secondly, RP has been worked out from which overcyclings are excluded and getting an answer about the class of the studed formula is guaranted during the smallest number of steps.

Thirdly, the cubical coverings make it possible to organize parallel pipelining (systolic processing) for them. A more detailed description of this question is given in [6], where the advantages of new class of processing elements – programmable systolic structures - are considered.

References

1. Вищняков В.А., Буланже Д.Ю., Герман О.В. Аппаратно-программные средства логического вывода. – М.: Радио и связь, 1991. – 264 с.

2. Robinson J.A. A Machine Oriented Logic Based on the Resolution Principle // J.ACM. - 1965.- 12. - P.23-41.

3. Chang C.L. and Lee R.C.T. Symbolic Logic and Mechanical Theorem Proving. - Academic Press, New York, 1973.

4. Семеренко В.П. Систолическая реализация кубических функций // Электронное моделирование. - 1992. – 1. - С.21-25.

5. Miller R. Switching Theory, vol.1: Combinational Circuits.- John Wiley and Sons, N.Y.,1965.

6. Семеренко В.П. Параллельная реализация логических рассуждений.- В сб. науч. трудов междунар. конф. "Знание-диалог-решение", В 2-х томах, т. 1. - Ялта, 1995, с.141-150.