A Constraint-Based Inference System for Satisfying Design Constraints

Joo-Heon Cha*
CAD/CAM Research Center at Korea Institute of Science and Technology
In-Ho Lee, Jay-Jung Kim
School of Mechanical Engineering at Hanyang University

We propose an efficient algorithm for the purpose of satisfying a wide range of design constraints represented with equality and inequality equations as well as production rules. The algorithm employs simulated-annealing and a production rule inference engine and works on design constraints represented with networks. The algorithm fulfills equality constraints through constraint satisfaction processes like variable elimination while taking into account inequality constraints and inferring production rules. It can also reduce the load of the optimization procedure if necessary. We demonstrate the implementation of the algorithm with the result on machine tool design.

Key Words : Inequality Constraints, Equality Constraints, Simulated-Annealing, Inference Engines, Intelligent CAD System, Production Rules

1. Introduction

The concept of a CAD system has progressively developed from a sophisticated drawing board into that of a design system that is capable of assisting designers throughout the design process. A recent CAD concept is the so-called intelligent CAD. Since a design process usually can't be completed in a few steps, the amount of design knowledge such intelligent CAD systems have to manage is inevitably huge(Zeiler, 1992). As a result, studies on intelligent CAD focus on how to represent and infer design knowledge effectively (Cha et al., 1993; Cha et al., 1994; Cha et al., 1998).

To manage the design knowledge, rule-based inference, object-oriented concept, constraint-programming techniques and data-handling techniques are studied and applied(Feng & Kusiak, 1995; Young et al., 1991). Among such techniques, constraint-programming techniques, in which design processes are considered as constraint satisfaction processes, have the advantage of declarative design knowledge expression (Murtagh & Masamichi, 1991; Yokoyama & Sazuka, 1990; Yoshihik, 1991). If knowledge is expressed declaratively, designers can handle massive data and access the data in a flexible, multi-perspective way. The constraint network is a collection of knowledge that is represented as constraints and connected by common variables (Serrano & Gossard, 1992; Thornton, 1993). In this paper, to manage various types of design knowledge, we have used constraint networks that are mainly used to manage equations.

When networks are used, representation and transformation, in particular, of design knowledge like equations are relatively easy. But if the equations to be satisfied have inequality constraints or they are under-constrained, it is not easy to perform constraint satisfaction completely. Furthermore, to represent knowledge of design objects properly, production rules are sometimes

* Corresponding Author,
E-mail : cha@kist.re.kr
TEL : +82-2-958-5648 ; **FAX** : +82-2-958-5639
CAD/CAM Research Center at Korea Institute of Science and Technology, P.O. Box 131, Cheongryang, Seoul 130-650, Korea. (Manuscript **Received** December 16, 1999; **Revised** February 18, 2000)

involved inevitably (Rich & Knight, 1992). Since such cases are frequently met in designing, a new algorithm is needed to overcome the situations mentioned. Until now, when the knowledge is inequality-constrained, the system turns over control to human designers. When the knowledge is under-constrained, the system uses extra optimization programs. When production rules are included in the knowledge, the system uses extra rule inference programs. Such systems inevitably have problems of inefficiency in supporting design processes and of burdening human designers with more work to do.

In this study, we propose a new constraint satisfaction algorithm that broadens the previous research work. As a knowledge representation method, we use networks that involve not only constraints such as equalities and inequalities but also production rules. It also fuses SA (simulated annealing; one of the powerful optimization algorithms) and the production rule inference to current constraint-based constraint propagation and the variable elimination method. This new algorithm can increase the efficiency of constraint satisfaction processes for several reasons: it reduces the load on optimization procedures and it performs production rule inferences, constraint satisfaction processes — such as constraint propagation and variable elimination — and the optimization process with SA simultaneously.

To show the effectiveness of the proposed constraint satisfaction algorithm, we implement an inference engine and insert the engine into the intelligent CAD system for machine tools design. In this study, the intelligent CAD system implemented is used to design basic structures of machining centers as an example.

2. Existing Methods to Represent and Satisfy Constraints

2.1. Constraint networks

2.1.1 Representation of design objects with constraint networks

Intelligent CAD systems must be able to store a huge amount of design knowledge for whole design processes and must be able to take human designers' place in design work and design supporting work. Accordingly, the representation methods for intelligent CAD systems must be standardized and efficient. To insure adequacy of the knowledge representation methods for such systems, the following four properties are needed (Rich & Knight, 1992):

- Representational adequacy: the ability to represent all kinds of knowledge that are needed in that domain;
- Inferential adequacy: the ability to manipulate the representational structures in such a way as to derive new structures corresponding to new knowledge inferred from the old ones;
- Inferential Efficiency: the ability to incorporate into the knowledge structure that can be used to focus the attention of inference mechanisms in the most promising directions;
- Acquisitional efficiency: the ability to acquire new information easily. The simplest case involves direct insertion, which put into a new knowledge into the database by a person;

As an adequate knowledge representation method, the constraint based representation method has been used in many fields of engineer ing design involving the CAD system. Constraint-based representation methods were used in electric circuit analysis at the beginning; after some studies on knowledge representation with constraints (Murtagh & Masamichi, 1991; Serrano & Gossard, 1992; Young et al., 1991; Cha et al., 1993), networks have been used in implementing design support systems.

Constraints provide an attractive way to represent many problems in AI; they provide a declarative formalism where one specifies what things should be without how to achieve this goal (Fromont & Sriram, 1992). In other words, with declarative representation, each design variable can play the same role without reforming and redefining constraints. For example, a design constraint "$F(a, b, c)=0$" of design variables "a", "b" and "c" can be constraints of other forms "$a=f_1(b, c)$", "$b=f_2(a, c)$" or "$c=f_3(a, b)$".

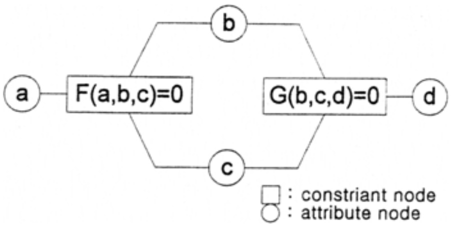


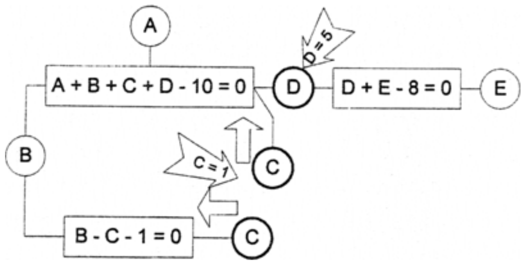
Fig. 1 A Simple example of the constraint network

Consequently, implementation or management of database systems and knowledge management systems can be considerably more easily facilitated using constraint representation methods. The constraint network is a graphical representation method of constraint representation as shown in Fig. 1. It shows a simple example of representing simple constraints. Here, circular nodes are attributes, square nodes are constraints, and arcs are their relations.

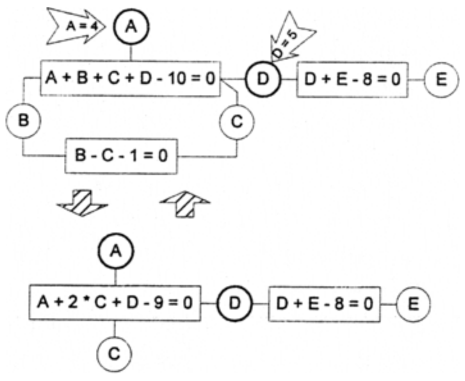
Some merits of using constraint networks are as follows. First, by virtue of declarative knowledge representation, usage of constraints can be flexible and effective. Consequently, the systems of constraint-based network can manage knowledge at low cost. Second, the systems can permit easy intervention of designers to implement inference systems that infer through human-computer interactions, and to introduce empirical knowledge or extra inference processes. Last, constraint satisfaction speed and accuracy are comparatively high (Cha et al., 1998; Serrano & Gossard, 1992; Thornton, 1993).

2.1.2 Constraint propagation and variable elimination

Previously, research has been performed to satisfy constraints that are represented with networks. The constraint propagation method proposed by Sussman G. J. & Steel G. L., Borning A., Murtagh N. & Shimura M. et al. is fit for intelligent CAD systems, but the constraint propagation method alone is not enough to solve problems when they are under-constrained or they involve closed-loops. To solve closed-loops of constraint networks, Serrano D., Fromont B. & Sriram D. et al. suggested to introduce optimization methods for the case when propagation is not



(a) When the close-loop is removed by user-defined variables



(b) When the closed-loop is found on constraint network

Fig. 2 Constraint satisfaction in constraint network

enough, but the number of variables for optimization can't be reduced completely. As another method to solve closed-loops, Kawamura T., Ohwada H. & Mizoguchi F. et al. introduced the variable elimination method, but it cannot give transparency and flexibility to intelligent CAD systems. Recently, Cha et al. proposed a new algorithm that can solve closed-loops and nonlinear problems by constraint propagation and variable elimination, and that represents complex design objects by modularization (Cha et al., 1993; Cha et al., 1994).

We use the algorithm that Cha et al. proposed as a basic constraint satisfaction method. The constraint satisfaction algorithm using constraint propagation and variable elimination is roughly separated into two ways depending on whether closed-loop exists or not. When no closed-loop is found on the networks, all the constraints can be satisfied with the constraint propagation method alone, but when any closed-loop is found on the networks, the algorithm is separated into two ways again depending on the user-defined node is on the closed-loop or not.

First, when any user-defined node is found on the closed-loop, after eliminating closed-loops by separating the user-defined node, constraint propagation is used to complete the satisfaction process. Figure 2(a) shows this case. In this case, after eliminating the closed-loop by separating the user -defined node "C", values of the nodes "A", "B" and "E" are calculated by constraint propagation.

Second, when no user-defined node is found on the closed-loops, after eliminating closed-loops by variable elimination, the constraint propagation method is used to complete the satisfaction process. Values of the eliminated variables are calculated after the constraint networks are backtracked. Figure 2(b) is showing this case. In this case, after breaking the closed-loop by eliminating the variable "B", values of the node "C" and "E" are calculated by constraint propagation. After variable elimination and constraint propagation processe by backtracking the network the value of node "B" can be calculated.

2.2 Production system

The production system is a knowledge representation method that is composed of independent rules represented as a series of "IF condition ~THEN process" (Rich & Knight, 1992). The production system is mainly used to represent empirical knowledge and has the following merits. First, since the production system is in the form of a common language, it is very easy to understand and use it. Second, since there is no relation between rules, fragmentary knowledge can be represented easily and adding and elimination of knowledge is facilitated. Third, since this is a very simple representation method, concrete knowledge about objects is not essential. Some weak points are inevitable in using the production system. First, inference can be infinitely repeated because rules have no relations. Second, inference can be inefficient because all rules must be considered though most of them don't have to be and because rules can be redundant and contradictary. Third, the results of inference have no guarantee, because knowledge conflict can accus. Thus, knowledge must be structuried and hierarchical.

2.3 Simulated-annealing

When a design process is highly standardized and concrete, constraints can be satisfied in a consistent manner. But otherwise, constraints must be satisfied by optimization considering constraints. In this study, therefore, an optimization algorithm is introduced for the case when problems are under-constrained and inequality-constrained.

Form numerous studies, various optimization algorithms have been developed and improved, but they sometimes fail to find optimal solutions. This makes designers try various starting points, search ranges, termination criteria etc. and select among the optimization results. Consequently, poor efficiency is a serious problem. To overcome this problem, GAs(Genetic Algorithms) and SA namely, "stochastic optimization algorithms" have been invented and used successfully in cases that are overly complex for other optimization algorithms and that have too many local optimum solutions (Laarhoven & Aarts, 1987; Man et al., 1996; Yang et al., 1999).

Both GA and SA have good search abilities, but have some different properties too. According to literature, GA can operate even though the objective functions are very vague, and it can search multi-points of multi-regions at one time, but needs particular setting like encoding to apply particular mathematical problems. On the other hand, SA can get comparatively accurate solutions when the objective functions are clear, and it is comparatively easy to apply mathematical problems because SA needs no additional process like encoding (Thornton, 1994). In this study, because our algorithm is for mathematical equations and production rules, SA is selected for the properties mentioned to apply under-constrained problems.

SA is a variant of the hill climbing method, which is an analogy to the physical phenomenon of annealing. Annealing refers to the process of metal at high temperature forming a specific molecular structure while slowly losing heat -energy. During this time, atoms move actively with high temperature before they form a specific structure at low temperature. SA makes variables search solution as atoms find a specific structure from various ones. To pattern after annealing, variables go under Metropolis' test following cooling schedule , as a result, probabilistic guarantee to arrive optimal solutions and to settle there, is given (Laarhoven & Aarts, 1987; Man et al., 1996; Yang et al., 1999).

Using SA, the following merits can be obtained. First, since SA is a powerful and general algorithm, it can be applied to various fields that need many things to consider such as VLSI design and image processing. Second, SA finds good solutions regardless of conditions and initial points. Third, since SA is a general optimization method and needs nearly no extra data like derivatives except objective functions, it is good to apply real engineering design that must consider various and complex conditions. Fourth, considering constraints is easy and intuitive in SA compared to other algorithms like the penalty method. On the other hend, the following can be cited as the demerits: SA needs comparatively long computing time and doesn't use declarative representation.

3. The Hybrid Type Constraint Satisfaction Method

3.1 The flow of the hybrid type constraint satisfaction method

The algorithm presented in this paper is designed to perform design processes through the following steps, to strengthen the merits and weaken the demerits by fusing the constraint-based inference algorithm, production rule inference, and SA. To make various inference processes be fused well and performed consistently, not only inequality constraints and objective functions but also production rules are considered as a sort of constraints, and involved in constraint networks. As a result, in the process of constraint satisfaction, if any constraint or variable does change, other constraints, variables, objective functions and production rules can immediately catch the change. In this way, a base on which various constraints can be considered simultaneously is prepared. Steps of the algorithm are as follows.

Step 0: Network-generation

Generate networks with received design constraints, to prepare constraint satisfaction processes.

Step 1: Design specifications input

Input design specifications adequate for design objectives

Step 2: Closed-loops elimination

Search closed-loops and eliminate it. The algorithm is separated into two ways again according as user-defined node is on the closed-loop. Here the closed-loops considered are made of equality constraints.

① When user-defined node is on the closed-loop, eliminate closed-loops by separating user-defined node.

② When user-defined node is not on the closed-loop, eliminate closed-loops by variable elimination.

Step 3: Constraint propagation and production rule inference

Perform production rule inference and constraint propagation.

Step 4: Optimization

If unconstrained constraints exist, satisfy the constraints by the SA method. In this study, during optimization process by SA, not only equality constraints and inequality constraints but also production rules are considered as a sort of constraints.

Step 5: Network backtracking

To satisfy constraints of the variables that are temporarily excluded by variable elimination, backtrack the transformed networks.

Step 6: Constraint satisfaction by constraint propagation and production rule inference

To satisfy all the constraints including eliminated variables, perform constraint propagation and production rule inference.

The flow chart of this algorithm including

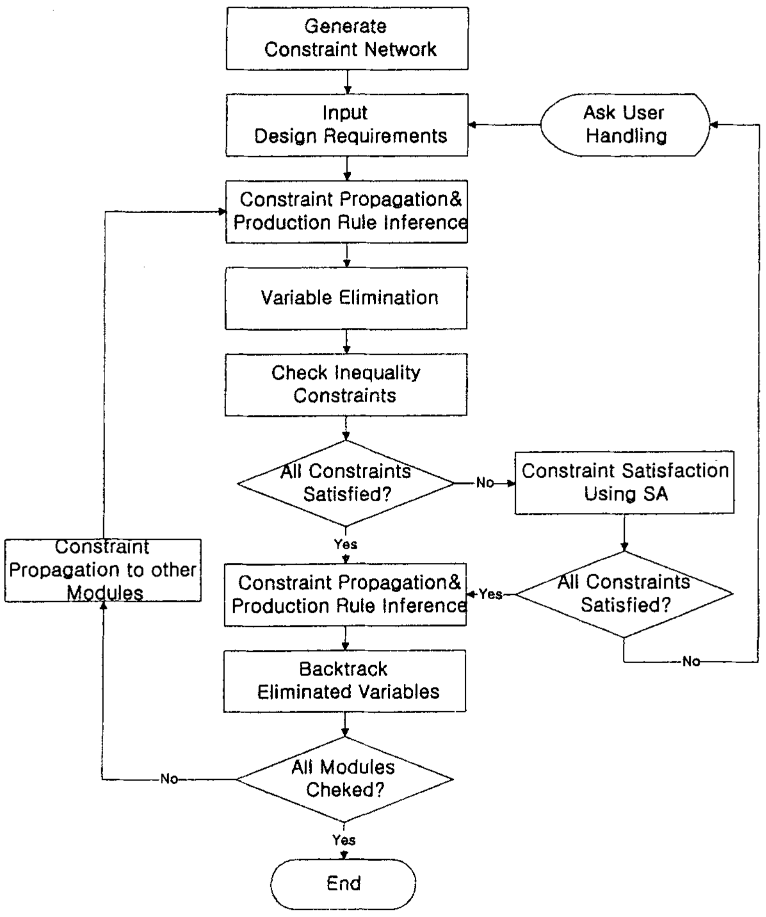

Generate Constraint Network
Input Design Requirements
Ask User Handling
Constraint Propagation& Production Rule Inference
Variable Elimination
Check Inequality Constraints
All Constraints Satisfied?
No
Constraint Satisfaction Using SA
Yes
Constraint Propagation to other Modules
Constraint Propagation& Production Rule Inference
Yes
All Constraints Satisfied?
No
Backtrack Eliminated Variables
All Modules Cheked?
No
Yes
End

Fig. 3 The flow chart of the hybrid constraint satisfaction algorithm

constraint satisfaction steps mentioned above is given in Fig. 3. Design knowledge is managed as constraint networks and separated into modules according to design flows and relations between knowledge. With the design specification put in, the hybrid-constraint satisfaction process, based on the constraint satisfaction steps, by variable elimination, constraint propagation, production rule inference and SA is performed. If any unconstrained variable exists, the user is asked to manipulate the situation. When all constraints are satisfied in a module, the constraints are propagated to other modules. Module after module, all constraints are satisfied.

3.2 Examples of constraint satisfaction with the hybrid-type algorithm

To explain the algorithm, we show two cases roughly separated by the condition, whether production rules to consider exist or not. In the following section 3.2.1 and 3.2.2, the two cases are explained with simple examples. Here, circular nodes with solid lines are variables, square nodes with solid lines are equality constraints, square nodes with dotted lines are inequality constraints and square nodes with black corners are production rules.

3.2.1 Constraint satisfaction without production rule inference

An example is prepared to explain this section. Constraints are as below, and a network made according to the constraints is shown in Fig. 4.:

- Equality constraints

$A+2BC+3D*D+4E-10=0$

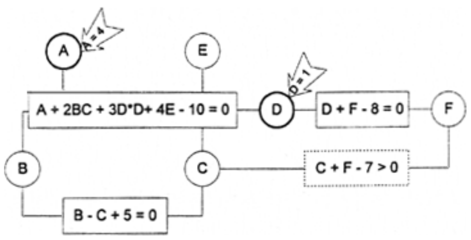


Fig. 4 A simple example for constraint satisfaction algorithm-1

$D+F-8=0$

$B-C+5=0$

■ Inequality constraints

$C+F-7>0$

■ Objective function

Min. $g_1(A, B, C, D, E)=3C^4-28C^3-168E+81A^*B+1600D$

Based on the algorithm, closed-loops are eliminated first. The network made of the example has closed-loops, and the user-defined nodes "A" and "D" are not on the closed-loops; therefore the loop is eliminated by the variable elimination method. The closed-loop made of equality constraints is removed by eliminating variable "B" using the constraint "$B-C+5=0$". After the elimination of the closed-loops, the constraints and objective function are transformed by propagating the user-defined variables "A" and "D" as below:

■ Equality constraints

$2C(C-5)+4E-3=0$

■ Inequality constraints

$C>0$

■ Objective function

Min. $g_2(C, E)=3C^4-28C^3+324C-168E-20$

The example of this section can be simplified by successive elimination of variable "E", with the constraints "$2C(C-5)+4E-3=0$" as below:

■ Inequality constraints

$C>0$

■ Objective function

Min. $g_3(C)=3C^4-28C^3+84C^2-96C-146$

Constraints which can't be satisfied by a chain of processes can be satisfied by optimization with the SA method. The value of "C" can be obtained as "$C=4$" after using the SA method.

Finally, the values of "B" and "E" that are hidden by the variable elimination method can be obtained after backtracking of the transformed networks. The results of all the satisfied constraints are as below:

$$B=0,\ C=4,\ E=0.5,\ F=7$$

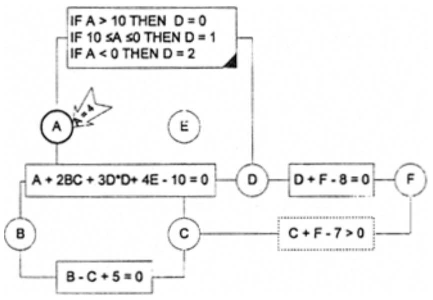


Fig. 5 A simple example for constraint satisfaction algorithm-2

3.2.2 Constraint satisfaction involving production rule inference

Two examples are prepared to explain this section. Figure 5, shown below, is the first simple example that has production rules. Since the network made of this example has a rule that has a fixed condition — the node "A" is a user-defined node — production rule inference can be performed as the first step. When the value of node "D" is obtained by production rule inference, the constraint satisfaction processes to be performed next are the same as the example in the previous section.

The second example of this section is shown below and the network made of the example is shown in Fig. 6. Since the network made of this example has a rule that has a non-fixed condition — the node "A" is not a user-defined node — production rule inference can't be performed completely before optimization. Therefore, production rule inference must be performed side by side with the optimization process of SA later.

■ Equality constraints

$A+2BC+3D^*D+4E-10=0$

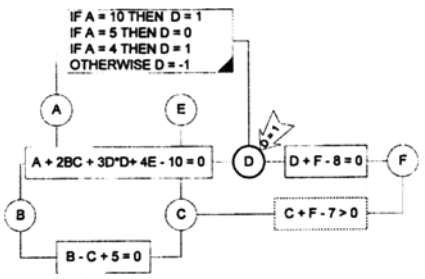


Fig. 6 A simple example for constraint satisfaction algorithm-3

$D+F-8=0$
$B-C+5=0$

▪ Inequality constraints
$C+F-7>0$

▪ Production rules
IF $A=10$ THEN $D=1$
IF $A=5$ THEN $D=0$
IF $A=4$ THEN $D=1$
OTHERWISE $D=-1$

▪ Objective function
Min. $g(A, B, C, D, E)=3C^4-28C^3-168D*E+324B+1600D$

As shown in Fig. 6, since the value of the only user-defined node "D" is not on the closed-loop which is made of equality constraints, the first step of the constraint satisfaction method is to remove the closed-loop by the variable elimination method. As a next step constraint propagation is performed. The result of variable elimination and constraint propagation is as shown below:

▪ Inequality constraints
$C>0$

▪ Production rules
IF $A=10$ THEN $D=1$
IF $A=5$ THEN $D=0$
IF $A=4$ THEN $D=1$
OTHERWISE $D=-1$

▪ Objective function
Min. $g_1(A, C)=3C^4-28C^3+84C^2+42A-314$

For the constraints to be satisfied last, the SA method that goes side by side with the production rule inference is performed. The result of SA and production rule inference is "$A=4$" and "$C=4$". Finally, the transformed network is backtracked and the values of the variables are obtained as follows:

$$A=4,\ B=-1,\ C=4,\ E=11/4,\ F=7$$

4. Implementation and Application of the Algorithm

To verify the proposed algorithm, we implement an intelligent CAD system to design machine tools, in which the algorithm is used. In addition, the intelligent CAD system is applied to design a machining center. The implementation is performed on an IBM PC with Visual c^{++}.

4.1 Implementation of an Intelligent-CAD system for machine tool design

To design machine tools is a highly intensive task, in which specialized and long-term design experience is essentially involved. In the design processes of machine tools, complex processes such as mechanical analysis, choice of standardized and non-standardized components, application of various sort of knowledge etc. are inevitably involved. Accordingly, if such complex and repeated tasks can be replaced by intelligent CAD systems, more efficient designs and shorter product development cycles can be obtained (Moriwaki et al., 1991; Moriwaki & Nunobiki, 1992).

Thus, in this study, an intelligent CAD system to design machine tools, more precisely machining centers (a type of machine tool), is implemented. The intelligent CAD system actually completes the machining center design through some successive processes as follows: First, taking the design knowledge such as production rules, equations and tables into consideration, the type of the machine tool is selected. After type selection, the geometric design process is performed from overall design to detailed design. At last, after analyses of the design results, redesign processes are performed.

The intelligent CAD system we have implemented has an inference engine that uses the proposed algorithm to conduct the complex design tasks. Figure 7 shows the diagram of the inference engine for the intelligent CAD system.

4.2 Structure-configuration design for machining center

The first step to design machining centers is to decide the type. To decide the type of machining center, according to the objects to be manufactured, a vertical type or a horizontal type is selected at first, and outlines of the structure are determined next. Then, a more detailed type is determined according to various special qualities of the types. As an example, part of the whole design knowledge to design the structure of machining centers is shown as in Fig. 8. Based on the knowledge, we decide types and compute dimensions of the table and the stroke. Here, the design knowledge data is composed of some keywords and some pieces of the design knowledge.

The constraint network, shown in Fig. 9 is made up of the design knowledge of Fig. 8. The intelligent CAD system waits for the design specification input after constructing networks as shown in the figure.

As the design specification of this example, the

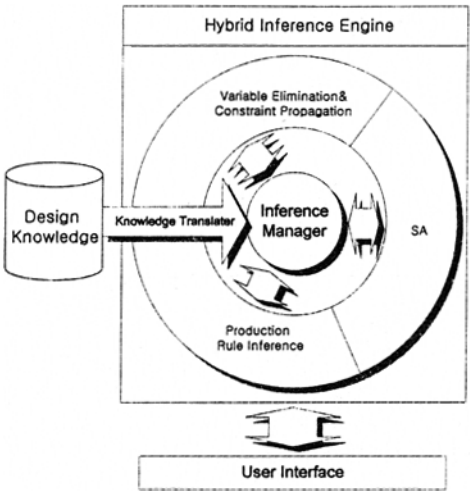


Fig. 7 Structure of the inference engine

```
OBJECT;
            Type&Table&Storke;
CONSTRAINT;
            Tl = TableGrade;
            Tw = Twratio * Tl;
            XStroke = Tw - 40;
            YStroke = Tl - 40;
            Zstroke = YStroke * StrRatio;
RULE;
            IF type = horizontal AND TableGrade = 400 THEN Th = 100;
            ELSE IF type = horizontal AND TableGrade = 500 THEN Th = 125;
            ELSE IF type = horizontal AND TableGrade = 600 THEN Th = 150;
            ELSE IF type = horizontal AND TableGrade = 700 THEN Th = 175;
            ELSE IF type = horizontal AND TableGrade = 800 THEN Th = 200;
            ELSE IF type = vertical AND TableGrade = 460 THEN Th = 45;
            ELSE IF type = vertical AND TableGrade = 480 THEN Th = 50;
            ELSE IF type = vertical AND TableGrade = 500 THEN Th = 50;
            ELSE IF type = vertical AND TableGrade = 520 THEN Th = 55;
            ELSE IF type = vertical AND TableGrade = 540 THEN Th = 55;
            OTHERWISE ERROR;
            IF type = horizontal THEN Twratio = 1;
            IF type = horizontal TEHN StrRatio = 1.5;
            ELSE IF type = vertical TEHN StrRatio = 1.1;
            OTHERWISE ERROR;
OBJECTIVE;
            MIN Tl*Tw*Th=0;
```

Fig. 8 part of the knowledge for machining center design

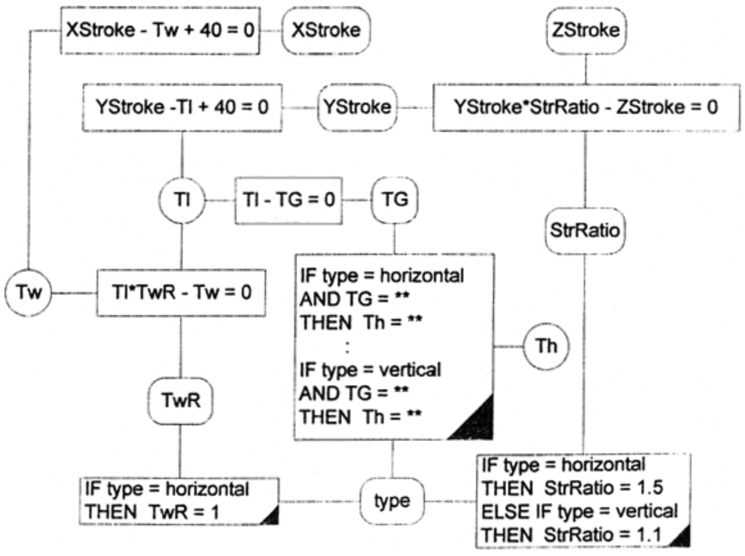

XStroke - Tw + 40 = 0
XStroke
ZStroke
YStroke -Tl + 40 = 0
YStroke
YStroke*StrRatio - ZStroke = 0
Tl
Tl - TG = 0
TG
StrRatio
Tw
Tl*TwR - Tw = 0
IF type = horizontal AND TG = ** THEN Th = ** : IF type = vertical AND TG = ** THEN Th = **
Th
TwR
IF type = horizontal THEN TwR = 1
type
IF type = horizontal THEN StrRatio = 1.5 ELSE IF type = vertical THEN StrRatio = 1.1

Fig. 9 Network of the knowledge for machining center design

Table 1 Results of machining center design

Variable		Value
table grade:	TG	500
table length:	Tl	500
table width:	Tw	1250
table height:	Th	50
table width ratio:	TwR	2.5
Stroke:	XStroke	1210
	YStroke	460
	ZStroke	506
type:	type	vertical

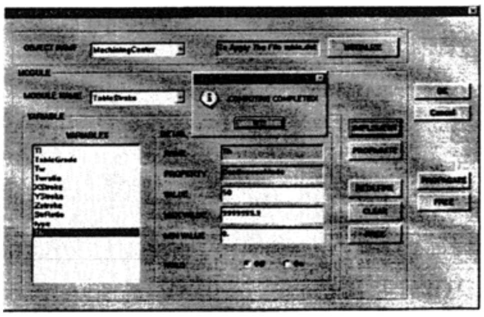


Fig. 10 Windown of inference engine

table grade "TG" is 500 and the table width ratio "TwR" is 2.5. By the algorithm proposed above, all constraints are satisfied and the values of all the design attributes are calculated. The design results of constraint satisfaction are shown in Table 1:

With the constraints and specifications mentioned above, the intelligent CAD system represents the design results when all the constraints are satisfied, are shown in Fig. 10.

In addition to the processes mentioned above, many other processes are performed in order to finish the whole machining center design. To show the designed machine tools graphically, they are represented by a 3-dimensional model. Figure 11 is the window showing the designed machine tool. The modeled machine tool in this section is a line-type vertical machining center. The cylinder attached to the backside of the machining center represents a motor and the two cylinders attached to the lateral side of the machining center represents a ATC and a Magazine.

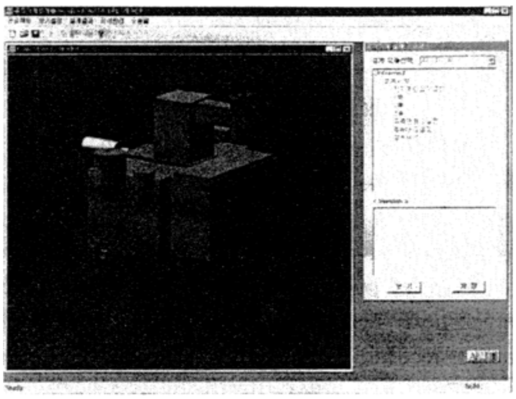


Fig. 11 Soild modeling window of line-type machining center

5. Conclusions

To support mechanical design tasks, in which many complex and repetitive processes exist, studies on intelligent CAD systems have been performed. In this study, constraint networks were used so that intelligent CAD systems were able to support design efficiently. Though the constraint networks are very effective and efficient as a representation method, efficient constraint satisfaction methods have not been invented yet for several cases as basted below:

- When inequality constraints are found on the networks;
- When the number of the constraints is not enough to be satisfied;
- When non-equation constraints as production rules exist.

This study one of the aims of is to develop an algorithm for the cases mentioned above. To begin with, we have regarded equality or inequality equations and production rules as a class of constraints. The networks, containing various constraints, are used in the new constraint satisfaction processes that we have proposed. In this algorithm, constraints are satisfied through a series of consistent steps composed of constraint propagation, variable elimination, production rule inference, and optimization.

Additionally, to show the effectiveness of the algorithm that is proposed here, we have implemented an intelligent CAD system for machine tool design using the algorithm. As a real design object, a machining center has been selected. Partrculorly in this study, the basic structure design process has been presented and designed.

In order to design complex mechanical products involving machine tools, in addition to the design knowledge types that are used in this study, other types of knowledge are needed (for example, the multi-variable that is a set of several rules to be satisfied at the same time). As a further study, a more powerful inference algorithm that uses a wider range of types of knowledge will be developed.

Reference

Cha, J. H., Yokoyama, M. and Okabe, I., 1993, "Constraint Satisfaction Method for Networks of Design Variables and Constraints in Mechanical CAD," *Trans. of JSME(C)*, Vol. 59, No. 568, pp. 3998~4003

Cha, J. H. and Yokoyama, M., 1994, "A Knowledge-Based System for Supporting Mechanical CAD," *Trans. of JSME(C)*, Vol. 60, No. 579, pp. 3625~3631.

Cha, J. H. Lee, I. H. and Kim, J. J., 1998, "Constraint Satisfaction System Considering Under-Constrained Problems," *8th Design & Systems Conference '98 JSME*, No. 98-32, pp. 354~357.

Feng, C. X. and Kusiak, A., 1995, "Constraint -based design of parts," *Computer Aided Design*, Vol. 27, No. 5, pp. 343~352.

Fromont, B. and Sriram, D., 1992, "Constraint Satisfaction as a Planning Process," in Gero, J. S. (ed.), *Artificial Intelligence in Design'92*, Kluwer Academic Pub, pp. 97~117.

Laarhoven, P. J. M. and Aarts, E. H. L., 1987, *Simulated-Annealing; Theory and Applications*, D. Reidel Publishing Co.

Man, K. F., Tang, K. and Kwong, S. S., 1996, "Genetic Algorithms: Concepts and Applications," *IEEE Transactions on Industrial Electronic*, Vol. 43, No. 5, pp. 519~534.

Moriwaki, T., Nunobiki, M., Nishimura, K., Yoshizawa, H. and Sakao, K., 1991, "A Study of Knowledge Based Approach to Basic Design of Machine Tools," *JSME(C)*, Vol. 57, No. 536, pp. 1371~1376.

Moriwaki, T. and Nunobiki, M., 1992, "Object -oriented Design Support System for Machine Tools," *JSME(C)*, Vol. 58, No. 546, pp. 655.

Murtagh, N. and Masamichi, S., 1991, "A Constraint-Based Hybrid Engineering Design Systems," *Intelligent CAD III*, IFIP, pp. 217 ~229.

Rich, E. and Knight, K., 1991, *Artificial Intelligence: Second Edition,* McGraw-Hill.

Salomons, W., Slooten, F., Houten, F. J. A. M., and Kals, H. J. J., 1995, "Conceptual Graphs in Constraint Based Re-Design," *Solid Modeling*, pp. 55~64.

Serrano, D. and Gossard, D. C., 1992, "Tools and Techniques for Conceptual Design," in Tong, C. and Sriram, D.(ed.), *Artificial Intelligence in Engineering Design*, Academic Press Inc., pp. 71 ~116.

Thornton, A. C., 1993, "Constraint Specification and Satisfaction in Embodiment Design," Cambridge University, Ph. D. thesis.

Thornton, A. C., 1994, "Genetic Algorithms Versus Simulated-Annealing: Satisfaction of Large Sets of Algebraic Mechanical Design Constraints," *Artificial Intelligence in Design* '94, Kluwer Academic Publishers, pp. 381~398.

Yang, B. S., Choi, B. G., Yu, Y. H. and Nan, H. T., 1999, "Optimum Design of a Damping Plate with an Unconstrained Viscoelastic Damping Layer Using Combined Genetic Algorithm," *KSME International Journal*, Vol. 13, No. 5, pp. 387~396

Yoshihiki, K., 1991, "An Approach to Specification Satisfaction by Network Based Method," *JSPE*, Vol. 57, No. 4, pp. 711~717.

Young, R. E., Greef, A. and O'Grady, P., 1991, "SPARK-An Artificial Intelligence Constraint Network System for Concurrent Engineering," in Gero, J. S.(ed.), *Artificial Intelligence in Design' 91*, pp. 79~94.

Zeiler, W., 1992, "Object-Oriented Hybrid Intelligent CAD System," *Computers in Industry 20*, Elsevier Science, pp. 1~9.