A knowledge-based parameter consistency management system for concurrent and collaborative design

Wei-ming Wang*, **Jie Hu**, **Ji-long Yin**, and **Ying-hong Peng**
School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, People's Republic of China

The manuscript was received on 17 August 2005 and was accepted after revision for publication on 13 September 2006.

DOI: 10.1243/09544054JEM450

Abstract: This paper discusses rule-based reasoning parameter consistency management system based on knowledge in concurrent and corroborative product design using a combination of both mathematical methods and knowledge-based Engineering (KBE) techniques. First, the framework of the parameter consistency management system is developed. Second, a data-mining algorithm known as the fuzzy–rough algorithm is presented for modelling of knowledge-level constraints from numerical simulation. Then, the method of integrated case-based reasoning (CBR) and rule-based reasoning (RBR) with an interval consistency algorithm is adopted to predict the potential conflicts and specify the interval of design parameters. Finally, ontology is used as a communication language between cooperating engineering design teams. The system is demonstrated by the design of a crank connecting link in a V6 engine.

Keywords: constraint network, knowledge-level constraints, fuzzy-rough algorithm, case-based reasoning, rule-based reasoning, consistency algorithm, engine

1 INTRODUCTION

Concurrent engineering (CE) is a systematic approach to integrating the design of products with related manufacturing processes using some software packages and computing techniques in a computer environment [**1**]. Within CE, a designer can consider and evaluate the downstream manufacturing processes of the product life cycle in the initial design phase. Collaborative design (co-design) is another increasingly important philosophy used in modern manufacturing corporations to collocate a multidisciplinary design team to carry out a complex design task through effective communication and collaboration. CE and co-design are complementary in functions since the former emphasises a vertically seamless linkage between the upstream design and the downstream manufacturing processes through the creation of intelligent strategies for effective information interchange, while the latter focuses more on the horizontally interpersonal aspects of group work in the upstream design phases [**2**].

In concurrent and collaborative design there are many constraints related to part features, feature–process relations, machine tools, cutting tools, cost, and time in product development [**3**]. Such constraints in the design process form a network of interconnected variables known as the constraint network which is a fairly recent development and can be applied to an extensive range of design and CE problems across a diverse range of industrial domains [**4**]. These networks offer a logical means of finding feasible solutions to complex design and planning problems by representing dependence relationships among the variables [**5**]. Oh *et al.* have thoroughly investigated the constraint network for concurrent design [**6**]. Other authors have developed several applicable constraint systems such as Saturn [**7**], Jupiter [**8**], and Spark [**9**]. A concurrent engineering oriented language based on the concept of the constraint network is investigated by Bowen and Bahler [**10**].

At the same time, computer aided engineering (CAE) technology has become the third mode of science complementing theory and experiment in concurrent and collaborative product design.

*Corresponding author: School of Mechanical Engineering, Institute of Knowledge Based Engineering, Shanghai Jiao Tong University, Dong-Chuan Road 800#, Shanghai 200240, People's Republic of China. email: wangweiming@sjtu.edu.cn

Knowledge-based engineering (KBE) is widely used in engineering, which integrates artificial intelligence with CAX systems and connects engineering design with CAX systems without interruption [**11**]. Numerical simulations in CAE generate vast quantities of data. These data can provide much useful knowledge, but they are simply stored away on disks and never analysed effectively. Therefore, extracting implicit knowledge as knowledge-level constraints from simulation results is very meaningful and urgent. However, modelling of knowledge-level constraints has not yet been fully addressed.

Knowledge discovery from databases (KDD) is a non-trivial process identifying valid, novel, potentially useful, and ultimately understandable patterns in data [**12**]. It can acquire implicit and useful knowledge in large-scale datasets and has had great success in commercial areas. It has expanded to engineering disciplines [**13**, **14**]. Although it is the success of KDD in the commercial area that is of interest in knowledge data based on simulation, there is a great difference between them.

To achieve consistent design parameters for all constraints in concurrent and collaborative design, an efficient parameter consistency management system should be provided for different design teams. The system should include a strategy for constraint conflict resolution to avoid disagreements within the different activities or domains [**15–18**]. Serrano and Gossard [**19**] developed a graphical theory approach to evaluating a set of constraints. Kannapan and Marshek [**20**] introduced a design diagram to represent parametric design problems and game theory to solve conflicts when constraints were violated. Danesh and Jin [**21**] take a decision-based approach to the model design process and introduce an agent-based decision network (ADN) to support concurrent decision-making and collaboration in design. Cooper and Taleb-Bendiab [**22**] describe the development of a framework for the support of multiparty negotiation for multiagent systems, which will be introduced through a general overview of the requirements of multiagent negotiation. Kusiak and Wang [**23**] proposed a reduction algorithm for qualitative constraint networks that can describe the effect of perturbances from one variable on the other variables. However, for some changes in the constraint network, such as the addition of new constraints, all of the relations between variables must be manually recalculated. Therefore, the static algorithm cannot adapt to new product development. Moreover, they did not propose a complete conflict resolution method incorporating mathematical constraints and knowledge-level constraints in concurrent and collaborative design.

None of these methods and tools can give a valid model to analyse all constraints in concurrent and collaborative design with a CAE system. Since knowledge-level constraints play an important role in engineering design, this paper describes a knowledge-based parameter consistency management system for concurrent and collaborative design using a combination of both mathematical methods and KBE techniques. The proposed method can combine the quantitative computation of feasible solution spaces with the rules from simulation and designers' expertise to determine design variables, which can reduce the multidisciplinary iterations in the concurrent design process. Section 2 describes the model of a parameter consistency management system. Section 3 introduces a data-mining algorithm known as the fuzzy–rough algorithm for acquisition of knowledge-level constraints from numerical simulation results. Section 4 describes the method of integrated case-based reasoning (CBR) and rule-based reasoning (RBR) with an interval consistency algorithm which is adopted to predict the potential conflicts and specify the interval of design parameters. Section 5 describes ontology used as a communication language between cooperating engineering design teams. Section 6 illustrates the design of a crank connecting link in a V6 engine using the proposed method. Section 7 concludes the paper.

2 SYSTEM FRAMEWORKS

2.1 Constraints in the design

Design variables in concurrent and collaborative product development are various, including symbol variables, logic variables, fuzzy variables, disperse variables, continuous variables, vectors, etc. Constraints in product development can be divided into the following four kinds according to the representation (Fig. 1).

1. *Equation constraints.* In these constraints, the relations between the variables are expressed by equalities, inequalities, and ordinary differential equation constraints, such as $L_2 = 2L_1$, $L_3 \geqslant L_1 + 2L_2$. Most of these constraints are geometry level, including the constraints between the part attributes, assembly dimension constraints, tolerance constraints, etc.
2. *Qualitative constraints.* Relations between the variables are expressed qualitatively, for example, type of oil pump = 'point blank'. Most of these constraints are product-level constraints, describing the general specification of product cost, performance, structure, production and maintenance, etc.
3. *Implicit constraints.* Black box constraints – there are certain relations between input and output, without knowing the detail in the black box.

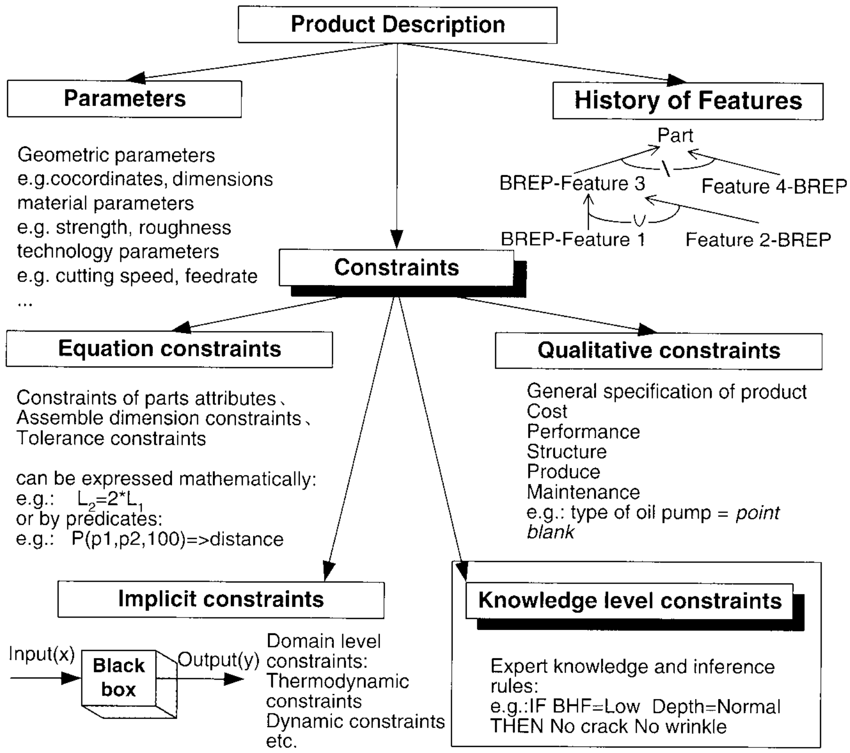

Fig. 1 Constraint-based product description

Most of these constraints are domain-level constraints, such as intensity constraints, dynamic constraints, thermodynamic constraints, etc.

4. *Knowledge-level constraints.* Knowledge-level constraints are implicit and useful constraints (knowledge) in large-scale simulation datasets and in the expert brain. The forms of these constraints are expert knowledge and the various inference rules between difference parameters and attributes during the design process. For example, 'IF $x_1 \leqslant x \leqslant x_2, y \leqslant y_1$, THEN $z_1 \leqslant z \leqslant z_2$' denotes quantified rules with the ability of numerical explanation, such as 'IF $5 \leqslant d \leqslant 25$, THEN $d_k = 4 + 0.52(d - 5)$'. A rule such as 'IF *a* AND *b*, THEN *c*' denote qualitative rules of reasoning, such as IF *BHF = low depth = normal*, THEN *no crack, no wrinkle*. Quantified rules are added to the (GDCN) to perfect the constraint network. Qualitative rules are added to the rule base for conflict resolution.

Knowledge-level constraints and other constraints can be united to be defined and managed by the GDCN.

Definition 1: Generalized dynamic constraint network

From the point of view of concurrent and collaborative design, GDCN is a constraint network considering all constraints in the product development. GDCN can be expressed as a set of equation constraints, qualitative constraints, implicit constraints, and knowledge-level constraints. The hybrid constraint based GDCN model can be

$$\boldsymbol{X} \in [\boldsymbol{X}^{\mathrm{L}0}, \boldsymbol{X}^{\mathrm{U}0}] \xrightarrow{A} \boldsymbol{X} \in [\boldsymbol{X}^{\mathrm{L}}, \boldsymbol{X}^{\mathrm{U}}] \tag{1}$$

Mathematic constraints

$$\boldsymbol{g}(\boldsymbol{Z}, \boldsymbol{X}) = 0, \quad \boldsymbol{g}_{\mathrm{T}}(\boldsymbol{X}, \boldsymbol{T}) = 0$$

$$\boldsymbol{h}(\boldsymbol{Z}, \boldsymbol{X}) \leqslant 0, \quad \boldsymbol{h}_{\mathrm{T}}(\boldsymbol{X}, \boldsymbol{T}) \leqslant 0$$

$$\frac{dZ}{dt} = f(\boldsymbol{Z}, \boldsymbol{X}, t) \tag{2}$$

Knowledge-level constraints

$$\text{IF } x_1 \leqslant x \leqslant x_2,\ y \leqslant y_1, \text{ THEN } z_1 \leqslant z \leqslant z_2$$

$$\text{IF } a \text{ AND } b, \text{ THEN } c \tag{3}$$

where $\boldsymbol{X} \in R^m$ is a generalized constraint variable vector, $[\boldsymbol{X}^{\mathrm{L}0}, \boldsymbol{X}^{\mathrm{U}0}]$ is the initial interval of generalized constraint variables, $[\boldsymbol{X}^{\mathrm{L}}, \boldsymbol{X}^{\mathrm{U}}]$ is the consistent interval of generalized constraint variables after treatment with algorithm A, $\boldsymbol{g}(\boldsymbol{x})$ is a constraint vector of equations, $\boldsymbol{h}(\boldsymbol{x})$ is an inequality vector, and f is an nth-order ordinary differential equation.

Knowledge-level constraints often hide in the bottom data. They cannot be mined and obtained through general methods. Therefore, developing effective knowledge-level constraints acquisition methods is the key to modelling a GDCN.

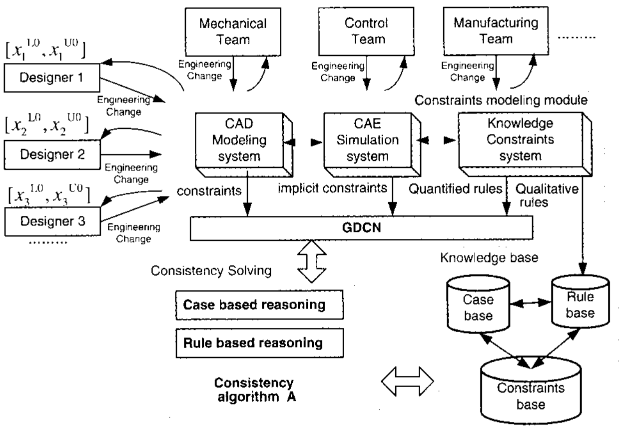

Fig. 2 Architecture of parameter consistency management system

2.2 Framework of the parameter consistency management system based on a GDCN

The proposed framework of parameter consistency management embodies a CAD solid modelling system, a CAE simulation system, a consistency manager system, and various knowledge bases. The architecture of the system is shown in Fig. 2.

As shown in Fig. 2, designers give the initial intervals $[\boldsymbol{X}^{L0},\boldsymbol{X}^{U0}]$ of design parameters. The first $[\boldsymbol{X}^{L0},\boldsymbol{X}^{U0}]$ may be large. Through modelling for all kinds of constraint from CAD modelling and CAE simulation, the GDCN is constructed as equations (1), (2), and (3). Then, conflict resolution strategy and algorithm A in (1) is used to check for consistency. Consistent intervals will be achieved after conflict resolution. Then, parameter intervals are consistent, and the GDCN will fix the intervals until the next engineering change.

1. *Constraint modelling module.* The constraint modelling module is used to model design parameters and requirements concerning various life-cycle issues. Constraints include variables of different design domains through CAD and CAE systems. Constraints are also collected from different knowledge sources such as experts, in which constraints can be formulated as rules, variables, values, and domains. These are equation constraints, qualitative constraints, implicit constraints, and knowledge-level constraints. The constraint modelling module is also linked to a consistency manager module.
2. *Consistency solving module.* Consistency solving is responsible for the management of the decision-making process and dealing with conflict situations. It detects conflicts and gives warnings and explanations to the users, finally applying a suitable strategy (CBR, RBR, or consistency algorithm) for solving conflicts in order to make sure that design consistency is in the constraint network and design output.
3. *Knowledge base.* The knowledge base includes the case base, rule base, and constraint base.

3 MODELLING OF KNOWLEDGE-LEVEL CONSTRAINTS

The geometric-level constraints in the equation constraints can be gained with parametric design and three-dimensional CAD software. There has been some progress in this field, especially concerning the variational geometric constraints and tolerance constraints [**24**, **25**]. A framework and computer implementation that provide automated methods for the specification of geometric tolerance types was presented in reference [**24**]. Moreover, based on the ISO/TC 213 geometric product specification, a new concept of variational geometric constraint network was presented. The generation and application of the network to specify geometric tolerance types was studied. At the same time, a computer aided approach of dimensional and geometric tolerance design was proposed in reference [**25**]. The method allows a designer to specify synthetically dimensional and geometric tolerances, including tolerance types and values. The application of the variation and tolerance constraints to specify tolerance values was studied.

A method for modelling implicit constraints can be constructed through abstracting relations between the input and the output (domain performance index) from simulation data based on the expand metamodelling method (EMM).

Knowledge-level constraints are complex and have various forms. Knowledge-level constraints often hide in the bottom data. They could not be mined through general methods to obtain rules that guide design. Therefore, developing effective domain-level constraint and knowledge-level constraint acquisition methods is the key to modelling the GDCN.

According to the characteristics of numerical simulation data, the process of knowledge-level constraint discovery from CAE result data is proposed as shown in Fig. 3.

Firstly, to study the relations between the design parameters and product performance, design of experiment (DOE) technology is used. The design structure is adjusted according to the experimental data. The simulation programs evaluate the performance of corresponding domains based on the adjusted design structure, such as the dynamic domain, the thermodynamic domain, the structural mechanics domain, etc. In the iterative process, large amounts of numerical simulation data related to design parameters are generated. These data are

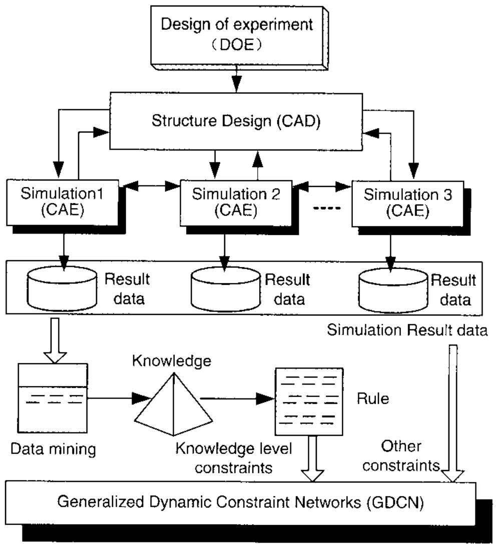

Fig. 3 Discovery processes of knowledge-level constraints from CAE simulation data

usually stored in flat files or special-format databases and can be used as a data source for knowledge-level constraint discovery.

The second step is data mining, an iterative process including five basic steps: domain understanding, data selection and integration, data preprocessing, rule induction, knowledge evaluation, and interpretation. In the domain understanding stage, the connotative meaning of every dataset and the mechanism by which they interact should be known clearly. The selected data will be used and analysed to give an answer to the problem under consideration. To improve the quality of the data for the dynamic modelling (DM) algorithm, data preprocessing must be done. In rule induction, intelligent methods are applied in order to extract data patterns. Production rules are selected as the knowledge representation form in this study owing to their modularity, simplicity, and expandability. The data mining process may be refined and some of its steps iterated several times before the extracted knowledge can be used.

The third step of the process is knowledge-level constraint management. Knowledge-level constraints by form of rule are added to the GDCN.

3.1 Acquisition algorithm

The rough-set theory (RST) proposed by Pawlak and Skowron has been widely used in knowledge reasoning and knowledge acquisition [**26**]. Since the basic RST algorithm can only handle nominal features in the decision table, most previous studies have simply shown how binary or crisp training data may be handled [**27**]. To apply the RST algorithm to real-value datasets, discretization often has to be applied as a preprocessing step to transform them into nominal feature spaces [**28**]. In the present study, an improved algorithm termed the fuzzy–rough set algorithm is developed by integrating fuzzy set theory with rough set theory. It can act as a DM algorithm in knowledge-level constraint discovery from numerical simulation result data.

It is very problematic that one object identified by a real-value attribute is exactly equal to another in the decision table. Therefore, equivalence relations in basic RST are too strict for quantitative data such as FEA simulation data. By introducing fuzzy indiscernibility relations to replace the equivalence relations in basic RST, the scope of information processing can be extended greatly. Also, the generated knowledge is nearer to natural language.

A decision table is $S = (U, A \cup \{d\})$. If V_a is composed of quantitative values, the value on attribute $a \in A$ can be catalogued into several fuzzy sets described by natural language such as 'low', 'normal', 'high', etc. Assume that the set L_a of linguistic terms of attribute a is $\{l^a_1, l^a_2, \ldots, l^a_{|L_a|}\}$. Object x belongs to the lth fuzzy set with fuzzy function f^x_{al}. For any two objects x and y, if there exists a linguistic term l of attribute a satisfying $f^x_{al} > 0$ and $f^y_{al} > 0$, it is said that there are fuzzy indiscernibility relations on single attribute a between objects x and y. The degree of indiscernibility on linguistic term l can be measured by $\mu_{al} = \min(f^x_{al}, f^y_{al})$. Similarly, if the same linguistic terms of an attribute subset B exist in both objects x and y with membership values larger than zero, x and y are said to have fuzzy indiscernibility relations on attribute subset B

$$IND'(B) = \{((x,y),\mu_B) : \forall_{a \in B}(f^x_{al} > 0, f^y_{al} > 0)\} \tag{4}$$

$[x]_{IND'(B)}$ denotes the fuzzy equivalence class of IND' (B) defined by x. Thus, fuzzy lower approximation and fuzzy upper approximation of subset X in U are defined as follows.

$$B'(X) = \{[x]_{IND'(B)}, \mu_B(x) : x \in U, [x]_{IND'(B)} \subseteq X\} \tag{5}$$

$$\overline{B'}(X) = \{([x]_{IND'(B)}, \mu_B(x)) : x \in U, [x]_{IND'(B)} \cap X \neq \Phi\} \tag{6}$$

By computing $B'(C_k)$ and $\overline{B'}(C_k)$ $(1 \leqslant k \leqslant r(d))$, certain and possible rules can be induced respectively. Also, the member value μ_B can be viewed as the efficiency measurement of the rule. This helps rule selection and sorting in knowledge reasoning. The mined rule set is usually redundant. Therefore, rule refinement must be made before use. The rule refinement criteria are listed as follows.

1. If the attribute description of one rule is more specific and a given measure is also lower than that of another, this rule can be removed from the rule set.

2. If the measurement of one rule is below some given threshold value, it should be removed from the rule set.
3. If one condition attribute is removed and collision occurs in the rule set, then this attribute should be removed.

In this way, an accurate and effective rule set can be obtained.

3.2 Detailed procedure of knowledge-level constraint discovery

Based on the above theory, the detailed steps of knowledge-level constraint discovery from numerical simulation data are summarized as follow.

Step 1. According to the domain knowledge, decide the centre point for fuzzy partition. Adopt a fuzzy member function to transform the quantitative value into several linguistic term descriptions.

Step 2. Compute the decision class C_k through $\{d\}$ (decision attribute subset).

Step 3. For any condition attribute subset $B \in \rho(A)$, compute the fuzzy equivalence class $IND'(B)$.

Step 4. For each decision class C_k, compute $B'(C_k)$ and $\overline{B'}(C_k)$ respectively, and insert them into a certain object set and uncertain object set respectively.

Step 5. Repeat steps 3 and 4 until all condition attribute subsets and all decision classes have been calculated.

Step 6. Certain rules are induced from certain object sets and the uncertain rules can be induced from uncertain object sets. Calculate the degree of support, accuracy, and efficiency measurement of each rule.

Step 7. Reduce the rule sets, and then add rules into the fuzzy rule knowledge base.

4 SOLVING FOR PARAMETER CONSISTENCY

Solving for parameter consistency is responsible for dealing with conflict situations. It detects conflicts, gives warnings and explanations to the users, and finally applies a suitable solving strategy and algorithm for solving conflicts in order to make sure that design consistency is in the GDCN and design output.

The process of solving for parameter consistency based on knowledge is shown in Fig. 4. Firstly conflicts are treated by the knowledge-based consistency solving module. If this subsystem can handle the conflict, RBR or CBR will be called to deal with the conflict; else conflicts will be transported to the algorithm-based consistency solving module. After solving successfully, cases and rules established during the solution process are added to the case base or rule base separately for perfecting the knowledge base.

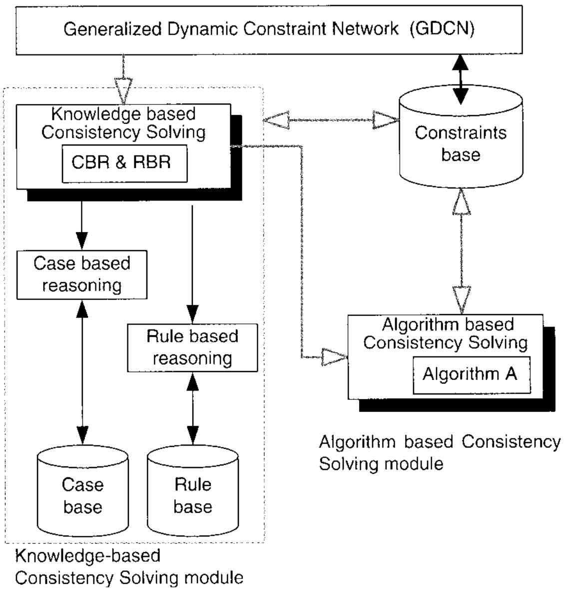

Fig. 4 Consistency solving integrated CBR and RBR with consistency algorithm

Procedure rule–reasoning (frame, slot):

BEGIN
Calling the centre reasoning module
REPEAT
Taking out a rule of conflict resolution from rules base: R_i;
Matching R_i;
IF (*matching successfully*)
THEN *executing rule conclusion*;
ELSE *continue*;
IF (*conflict solved successfully*)
THEN *stopping rule reasoning*;
ELSE *extending rules;*
IF (*no rules can be used*)
THEN *stopping rule reasoning*;
UNTIL *no rule could be matched*
END

In the algorithm-based consistency solving module, the consistency algorithm determines the interval of design parameters.

Definition 2, Conflict

If all intervals in vector $[\boldsymbol{x}^L, \boldsymbol{x}^U]$ are empty after being refined using the algorithm in equation (1), the present constraint network model contains a conflict, which means that no solution exists in the present constraint network model and the design is unfeasible.

Definition 3, algorithm A

Algorithm A can be expressed as

$$\boldsymbol{x} \in [\boldsymbol{x}^{\mathrm{L0}}, \boldsymbol{x}^{\mathrm{U0}}] \xrightarrow{A} \boldsymbol{x} \in [\boldsymbol{x}^{\mathrm{L}}, \boldsymbol{x}^{\mathrm{U}}]$$

so that

$$\boldsymbol{g}(\boldsymbol{x}) = 0,\ \boldsymbol{g}(\boldsymbol{x}) = [\boldsymbol{g}_1(\boldsymbol{x}), \boldsymbol{g}_2(\boldsymbol{x}), \ldots \boldsymbol{g}_l(\boldsymbol{x})]^{\mathrm{T}}$$

$$\boldsymbol{h}(\boldsymbol{x}) \leqslant 0,\ \boldsymbol{h}(\boldsymbol{x}) = [\boldsymbol{h}_1(\boldsymbol{x}), \boldsymbol{h}_2(\boldsymbol{x}), \ldots \boldsymbol{h}_m(\boldsymbol{x})]^{\mathrm{T}} \tag{7}$$

where $\boldsymbol{x} \in \boldsymbol{R}^n$ is the nth dimensional design parameter vector, g and h are the equation and inequation constraint functions for the parameters, $[\boldsymbol{x}^{\mathrm{L0}}, \boldsymbol{x}^{\mathrm{U0}}]$ is the original evaluated parameter interval, and $[\boldsymbol{x}^{\mathrm{L}}, \boldsymbol{x}^{\mathrm{U}}]$ is the consistent interval filtered by the consistency algorithm that corresponds to the given design goals.

Definition 4, Consistency

If all of the intervals have been refined using algorithm A' and no conflict is found, then the refined intervals in the constraint network model achieve consistency.

To keep uniformity between all of the constraints, a secondary variable vector $\boldsymbol{\theta} \in R^q(\boldsymbol{\theta} = 0)$ is added to translate inequation to equation:

$$\boldsymbol{h}(\boldsymbol{x}) \leqslant 0 \Rightarrow \boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{\theta} = 0 \tag{8}$$

By equation (8), equation (7) is transformed into

$$\boldsymbol{x} \in [\boldsymbol{x}^{\mathrm{L0}}, \boldsymbol{x}^{\mathrm{U0}}] \xrightarrow{A} \boldsymbol{x} \in [\boldsymbol{x}^{\mathrm{L}}, \boldsymbol{x}^{\mathrm{U}}]$$

so that

$$\boldsymbol{g}(\boldsymbol{x}) = 0, \boldsymbol{g}(\boldsymbol{x}) = [\boldsymbol{g}_1(\boldsymbol{x}), \boldsymbol{g}_2(\boldsymbol{x}), \ldots, \boldsymbol{g}_{l+m}(\boldsymbol{x})]^{\mathrm{T}} \tag{9}$$

The relationship matrix is defined to describe the relationship between the constraints and the variables in the constraint network model described in equation (9), and is defined as

$$\boldsymbol{G} = \{\boldsymbol{g}(ij)\}, \qquad \boldsymbol{g}(ij) = \begin{cases} 1, \boldsymbol{g}_i \text{ contains } \boldsymbol{x}_j \\ 0, \boldsymbol{g}_i \text{ does not contain } \boldsymbol{x}_j \end{cases} \tag{10}$$

where $1 \leqslant i \leqslant l+m, 1 \leqslant j \leqslant k+m$.

In addition, two secondary vectors are defined

$$\boldsymbol{p}_c = [\boldsymbol{p}_c(1), \boldsymbol{p}_c(k+m)]$$

$$\boldsymbol{p}_c(\lambda) = \begin{cases} 1, \lambda = c \\ 0, \lambda \neq c \end{cases} \qquad c = 1, 2, \ldots, k+m \tag{11}$$

$$\boldsymbol{q}_r = [\boldsymbol{q}_r(1), \boldsymbol{q}_r(2), \ldots, \boldsymbol{q}_r(k+m)]^{\mathrm{T}} \tag{12}$$

In Equation (9), making $\boldsymbol{p}_i\boldsymbol{G}\boldsymbol{q} \neq 0$ for every j gives the solution functions of constraint $\boldsymbol{g}_j$, which are

$$\boldsymbol{x}(j) = \boldsymbol{g}_i^j(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{j+1}, \ldots, \boldsymbol{x}_{k+m})$$

where $\boldsymbol{g}_i^j$ is actually the projection of constraint $\boldsymbol{g}_i$ on variable $\boldsymbol{x}(j)$, $1 \leqslant j \leqslant k+m$.

The paper uses a consistency algorithm to solve the interval problem described by equation (9). The consistency algorithm uses the following notations.

Algorithm A:

```
  Procedure LabelResolve (In {C = [g_i^j], x = 0,
y = [x^L0, x^U0]}
    BEGIN
      WHILE (y ≠ x) DO
      x = y
      FOR j = 1 to k + m
      FOR i = 1 to l + m
      IF (p_i G q_j ≠ 0)
      y_i = g_i^j(x)
      ENDIF
      IF y_i = = {null set}
      Show conflict message
      Exit
      ENDIF
      END FOR
      END FOR
  ENDWHILE
  END
  END Procedure
```

5 ONTOLOGY-BASED CONSTRAINT INFORMATION SHARING

For generalized constraints, the conventional expressions usually leave implicit details to be understood. Whether the magnitudes are vectors or higher-order tensors, the user must interpret the constraint notation using background knowledge and context. This is error prone for networked design and manufacturing.

To enable the sharing and reuse of constraints and parameter consistency information among engineering tools and their users, it is important to specify a conceptual foundation that makes these distinctions explicit and provides context- and reader-independent semantics. An ontology-based method was developed for constraint network modelling in concurrent and collaborative design for networked manufacturing. Ontology provides a sharable representation of knowledge that minimizes ambiguity and maximizes understanding and precision in communication.

To illustrate the use of ontology, consider a simple example exchanging symbolic representations of geometric constraints. Agent A is a specialist in the design of geometric parameters, and agent B is a specialist in optimization. Agent A needs a solution to a set of equations that include the following

$$\sigma \geqslant \frac{c_1}{\pi f_{\mathrm{v}} f_{\mathrm{w}} my.x_6} \tag{13}$$

where f_v and f_w are velocity load factors, σ is the allowable stress of the gear, m is a module, y is the form factor, a is the gear ratio, x_6 is the geometric dimension, and c_1 is a constant.

The ontology is expressed as follows.

(scalar-quantity Sigma)
(= >(physical.dimension Sigma)
(/force-dimension area-dimension))
(scalar-quantity fv)
(= (physical.dimension fv) velocity factor
-dimension)
(scalar-quantity fw)
(= (physical.dimension fw)load factor
-dimension)
(scalar-quantity m)
(= (physical.dimension m) module-dimension)
(scalar-quantity y)
(= (physical.dimension y) form
factor-dimension)
(scalar-quantity x6)
(= >(physical.dimension x6)
(/force-dimension stress-dimension))
(= >Sigma (/(c1)(* 3.14159 (fv)(fw) (m)(y)(x6))))

Agent A can send agent B these equations as a set of sentences, using the vocabulary of the ontology.

The following message is sent to agent B in ontology format.

(evaluate : sender agent-a
: receiver agent-b
: language ontology
: content ()
: reply-with mess-1)
The following message is sent in reply:
(reply : sender agent-b
: receiver agent-a
: language ontology
: content ()
: in-reply-to mess-1)

In this paper, example agents have been described: kinematic agent, dynamics agent, knowledge acquisition agent, and consistency manager agent.

1. The kinematic agent is based on three-dimensional CAD software and provides geometric, variation geometric, and tolerance constraints.
2. The dynamics agent is based on dynamics software and provides other implicit discipline constraints.
3. The knowledge acquisition agent is based on CAE software and experts to abstract the implicit design and manufacture rules.
4. The consistency manager agent is based on CBR, RBR, and an interval consistency algorithm to predict the potential conflicts and specify the interval of design parameters.

6 DESIGN EXAMPLES

The crank and connecting rod design of a V6 engine (shown in Fig. 5) is used as an example to show the validity of the knowledge-based constraint network system (KCNS) in concurrent and collaborative design. In fact, crank and connecting rod design is a collaborative design with a multidomain. The

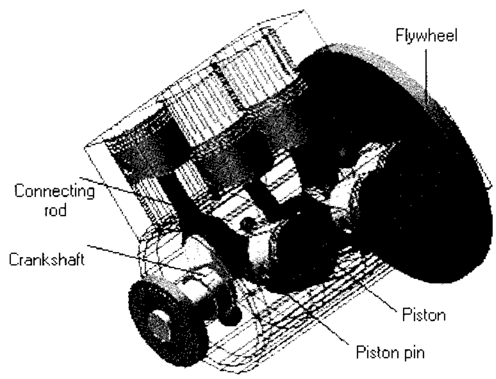

Fig. 5 Crank and connecting rod configuration of a V6 engine

Table 1 Rules considered

Rule	Effectiveness (%)
a2 = 0⇒a3 = 2	48.15
a0 = 0 a2 = 1⇒a3 = 1	7.41
a1 = 1 a2 = 2⇒a3 = 0	11.11
a0 = 0 a2 = 0⇒a3 = 2	25.93
a1 = 2 a2 = 2⇒a3 = 0	7.41
...	...
a1 = 1 a2 = 2⇒a3 = 0	7.41
a1 = 0 a2 = 0⇒a3 = 2	29.63
a2 = 1⇒a3 = 0	22.22
a1 = 1 a2 = 1⇒ a3 = 0	7.41

Table 2 Induced 'What-if' rules

Rules	Effectiveness (%)
IF *mass* = *low length* = *normal diameter* = *low* THEN *no crack*	69
IF *mass* = *normal length* = *high diameter* = *normal* THEN *no crack*	88
...	...
IF *mass* = *normal length* = *high* THEN *no crack*	100
IF *mass* = *high length* = *normal diameter* = *normal* THEN *no crack*	86
IF *length* = *high diameter* = *low* THEN *no crack*	69
IF *mass* = *high length* = *low diameter* = *low* THEN *crack*	67

Table 3 Design constraints

Constraint name	Constraint expression	Responsibility
Reciprocate inertia force of con-rod, p_{c1}	$p_{c1} = -mca$	Mechanical design team
Centrifugal inertia force of con-rod, p_{c2}	$P_{c2} = -m_c l_a \lambda^2 \omega^2 \cos^2\alpha(1-\lambda^2 \sin^2\alpha)$	Mechanical design team
Palstance inertia force of con-rod, M_c	$M_c = I_c \lambda(1-\lambda^2)\sin\alpha(1-\lambda^2\sin^2\alpha)^{3/2}$	Mechanical design team
Velocity of piston, ν	$\nu = r\omega\sin(\alpha+\beta)/\cos\alpha$	Mechanical design team
Palstance of con-rod, ω_1	$\omega_1 = \omega\lambda\cos\alpha/(1-\lambda^2\sin^2\alpha)^{1/2}$	Mechanical design team
Displacement of plunger, x	$x = (r+l)-(r\cos\alpha + l\sin\alpha)$	Mechanical design team
Allowable fatigue stress, σ_{max}	$\sigma_{max} \leqslant \sigma_u/[n]$	Mechanical design team
Deformation of stiffness, ΔR_{max}	$\Delta R_{max} < D_1/2$, $\Delta r_{max} < D_2/2$	Mechanical design team
Boundary constraint	$R_1-d>0$, $R_2-D>0$	Mechanical design team
Implicit constraints	$f_1(x) = 34647-837.88x_1+13.96x_2-16.84x_3+3.37x_1x_2+37.35x_2x_3-1.669x_1x_3$	CAE team
Knowledge-level constraints	IF 20 (mm) < diameter $\leqslant$ 20.1(mm), THEN force $\leqslant$ 34 108 (N)	CAE and KBE team
Knowledge-level constraints	IF *Mass* = *low length* = *normal diameter* = *low* THEN *no crack*	CAE and KBE team

Table 4 Consistency result from the constraint network

Symbol	Description	x^{L0}	x^{U0}	x^L	x^U
D (*mm*)	Diameter of cylinder	65.00	75.00	68.00	73.50
$1/\lambda$ (*mm*)	Ratio of connecting rod	3.20	3.80	3.40	3.60
m (*kg*)	Mass of connecting rod	0.59	0.63	0.60	0.61
L (*mm*)	Length of connecting rod	50.00	60.00	55.37	56.45
d_1 (*mm*)	Diameter of pin in the little end	19.50	30.00	24.50	26.00
d (*mm*)	Diameter of pore in the little end	16.25	22.50	18.00	21.60
D_1 (*mm*)	Form dimension of little end	17.00	25.90	20.80	22.25
B_1 (*mm*)	Breadth of little end	20.50	34.90	24.70	30.06
D_2 (*mm*)	Diameter of pin (crankshaft-butt)	15.50	23.60	18.90	20.08.
B_2 (*mm*)	Length of pin in the big end	40.90	52.50	46.40	47.08
D_2 (*mm*)	Diameter of pore in the big end	24.50	36.80	28.40	30.05
d_m (*mm*)	Diameter of bolt hole in big end	53.00	71.25	56.20	58.03
R_1 (*mm*)	Radius of knuckle in little end	7.80	10.50	9.30	10.02
R_2 (*mm*)	Radius of knuckle in big end	58.00	68.00	60.10	62.05

design involves all kinds of constraint, such as equation constraints, qualitative constraints, implicit constraints, knowledge-level constraints.

The engine parameters are as follows: number of cylinders $n = 6$; firing order (n_ignite) 1–4–2–6–3–5; balance layout (f_balwt) 1-3-6-7-10-12; number of revolutions 3000 r/min; material of connecting rod 40Mn2S.

The multi-body dynamics software package ADAMS is selected as a simulation tool. The parameters of interest in design are the mass of the connecting rod (mass), the length of the plug between the basic shaft and the connecting rod (length), and the diameter of the plug between the basic shaft and the connecting rod (diameter). They are denoted by a_0, a_1, and a_2 respectively. The force of the crankshaft (force) is a goal attribute. It is denoted by a_3. A DOE method known as the orthogonal Latin square is adopted. The simulations are carried out 64 times with various design parameters.

As shown in Table 1, according to the proposed method of knowledge-level constraint acquisition, 16 quantified rules are discovered. Some of these rules are as follows: 'a2 = 0 ⇒ a3 = 2' expresses IF diameter $\leqslant$ 20(mm), THEN force > 34 110N; 'a0 = 0 a2 = 1⇒a3 = 1' expresses IF mass $\leqslant$ 0.60500 kg, 20 mm < diameter $\leqslant$ 20.10 mm, THEN 34 108N <force $\leqslant$ 34 110 N; 'a2 = 1 ⇒ a3 = 0' expresses IF 20 mm < diameter $\leqslant$ 20.1 mm, THEN force $\leqslant$ 34 108N.

As shown in Table 2, according to the proposed method of knowledge-level constraint acquisition, 14 qualitative rules are discovered.

The following rule is selected for illustration: 'IF mass = low AND length = normal AND diameter = high, THEN no crack ($R = 69$ per cent, $P = 100$ per cent, $S = 1$)'. This means that, when the mass of the connecting rod is low, the length of the plug between the basic shaft and the connecting rod is normal, and the diameter of the plug between the basic shaft and the connecting rod is low, no cracking occurs in the crankshaft. The effectiveness of the rule is 100 per cent, which means that the rule has a 69 per cent tolerance (or effectiveness) on the data. It is helpful to schedule this measure in future knowledge reasoning. $S = 1$ indicates that there is only one conditional attribute in the priors of this rule, so it is very abstract and concise.

All constraints of crank and connecting rod design that are considered during the whole design process are listed in Table 3.

Table 4 lists the consistency results caused by an engineering change in the design process. The input of the constraint network consists of design parameter intervals given by designers using their experience. The output consists of filtered design intervals in which conflict parameters have been removed.

Conflicts occur when no solutions exist within the given bounds that satisfy all of the constraints. The parameter must then be revised until the consistency checking is satisfied. The design can be viewed as a process represented by a series of engineering changes in the constraint network. In Table 4, the upper and lower estimate limits are $\boldsymbol{x}^{L0}$ and $\boldsymbol{x}^{U0}$ respectively. After being refined, they are identified as $\boldsymbol{x}^{L}$ and $\boldsymbol{x}^{U}$, as in equation (9).

When the design is completed, the consistency design intervals are $L \in [55.37, 56.45]$, $m \in [0.60, 0.61]$, $D_2 \in [18.90, 20.08]$, etc.

The crankshaft design is a typical concurrent and collaborative design problem. The current method achieves a satisfactory solution for the parameter. Although the example is simplified, the methodology can be applied to actual design. The method described in this paper was used to develop a concurrent and collaborative design system using Visual.NET, which is now running in an engine design project.

7 CONCLUSIONS

This paper describes a knowledge-based parameter consistency management system for concurrent product design using a combination of both mathematical methods and knowledge-based engineering (KBE) techniques. First, a model of knowledge-based parameter consistency management system is developed. Second, a data-mining algorithm termed the fuzzy–rough algorithm is presented for modelling knowledge-level constraints from numerical simulation results. Third, the method of integrated CBR and RBR with an interval consistency algorithm is adopted to refine the intervals. Finally, ontology is used as a communication language between cooperating engineering design teams. The system is demonstrated by the design of a crank connecting link in a V6 engine.

ACKNOWLEDGEMENTS

This paper is supported by the National Natural Science Foundation of China (No. 60304015 and 50575142), Postdoctoral Science Foundation of China (No. 20060400646), and the Natural Science Foundation of Shanghai, China (No. 04ZR14081).

REFERENCES

1 **Ranky, P. G.** Current/simultaneous engineering: methods, tools and case studies, 1994 (CIMware Limited, Guildford, UK).

2 **Lia, W. D., Fuh, J. Y. H.,** and **Wong, Y. S.** An Internet-enabled integrated system for co-design and concurrent engineering. *Computers in Industry*, 2004, **55**, 87–103.

3 **Xiong, G. L.** and **Chang, T. Q.** Coordination model for the concurrent engineering product development process. *High Technol. Lett.*, 1998, **4**(2), 1–8.

4 **Abdalla, H. S.** A concurrent engineering constraints-based system. *Computers in Engng*, 1998, **3**(35), 459–462.

5 **Young, R. E., Giachetti, R. E.,** and **Ress, D. A.** A fuzzy constraint satisfaction system for design and manufacturing. In Proceedings of FUZZ – IEEE. 1996, vol. II, pp. 1106–1112.

6 **Oh, J. S., O'Grady, P.,** and **Young, R. E.** An artificial intelligence constraint network approach to design for assembly. *IIE Trans. (UK)*, 1995, **1**(27), 72–80.

7 **Young, R. E., Greef, A.,** and **O'Grady, P.** An artificial intelligence-based constraint network system for concurrent engineering. *Int. J. Prod. Res.*, 1992, **7**(30), 1715–1735.

8 **Fohn, S., Greef, A., Young, R. E.,** and **O'Grady, P.** A constraint-system shell to support concurrent engineering approaches to design. *Artif. Intell. Engng*, 1994, **9**, 1–17.

9 **Liau, J., Young, R. E.,** and **O'Grady, P.** Combining process planning and concurrent engineering to support printed circuit board assembly. *Comput. Ind. Engng*, 1995, **3**(28), 615–629.

10 **Bowen, J.** and **Bahler, D.** Frames, quantification, perspectives, and negotiation in constraint network for life-cycle engineering. *Artif. Intell. in Engng* 1992, **7**, 199–226.

11 **Wallace, G. S.** Introduction of knowledge based engineering for design for manufacture in the automotive industry. In Proceedings of the 1997 Conference on *Successful cases of integrated product design with manufacturing technology*, London, May 1997.

12 **Fayyad, U., Piatetsky-Shapiro, G., Smyth, P.,** and **Uthurusamy, R.** Advances in knowledge discovery in databases, 1996 (MIT Press, Cambridge, Massachusetts).

13 **Grossman, R. L.** (Ed.) Data mining for scientific and engineering applications, 2001 (Kluwer Academic Publishers, Dordrecht, Netherlands).

14 **Han, J.** and **Kamber, M.** Data mining: concepts and techniques 2001 (Morgan Kaufmann Publishers, San Fransisco).

15 **Noble, J. S.** An approach to overall design consistency in concurrent engineering. *Int. J. Flexible Autom. Integrated Mfg*, 1993, **1**(1), 25–36.

16 **Balasubramanian, S.** and **Norrie, D. H.** A multi-agent architecture for concurrent design, process planning, routing, and scheduling. *Concurrent Engng Res. Appl.* 1996, **4**(1), 7–15.

17 **Lander, S. E.** and **Corkill, D. D.** Designing integrated engineering environments: blackboard-based integration of design and analysis tools. *Concurrent Engng Res. Appl.*, 1996, **4**(1), 59–70.

18 **O'Grady, P., Young, R. E., Greef, A.,** and **Smith, L.** An advice system for concurrent engineering. *Int. J. Computer Integrated Mfg.*, 1991, **4**(2), 63–70.

19 **Serrano, D.** and **Gossard, D.** Tools and techniques for conceptual design. *Artificial intelligence in engineering*

design (Ed. A. Kusiak) 1992, pp. 71–116 (Wiley, New York).

20 **Kannapan, S. M.** and **Marshek, K. M.** An approach to parametric machine design and negotiation in concurrent engineering. *Concurrent engineering: automation, tools and techniques* (Ed. A. Kusiak), 1993, pp. 509–533 (Wiley, New York).

21 **Danesh, M. R.** and **Jin, Y.** An agent-based decision network for concurrent engineering design. *Concurrent Engng Res. and Applic.*, 2001, **1**(9), 37–47.

22 **Cooper, S.** and **Taleb-Bendiab, A.** CONCENSUS: multi-party negotiation support for conflict resolution in concurrent engineering design. *J. Intell. Mfg*, 1998, **2**(9), 155–159.

23 **Kusiak, A.** and **Wang, J.** Dependency analysis in constraint negotiation. *IEEE Trans. Syst., Man. Cybern.*, 1995, **25**, 1301–1313.

24 **Hu, J., Xiong, G.,** and **Wu, Z.** A variational geometric constraints network for a tolerance types specification. *Int. J. Advd Mfg Technol.*, 2004, **24**, 214–222.

25 **Hu, J.** and **Xiong, G.** Dimensional and geometric tolerance design based on constraints. *Int. J. Advd Mfg. Technol.*, 2005, **26**(9), 1099–1108.

26 **Pawlak, Z.** and **Skowron, A.** Rough set rudiments. *Bull. Int. Rough Set Soc.*, 1999, **4**(3), 181–185.

27 **Zhong N., Dong, J. D., Ohsuga, S.,** and **Lin, T. Y.** An incremental probabilistic rough set approach to rule discovery. *IEEE International Conference,* 1998, pp. 138–149.

28 **Dougherty, J., Kohavi, R.,** and **Sahami, M.** Supervised and unsupervised discretization of continuous features. In Proceedings of ICML-95, 12th International Conference on Machine Learning, Lake Tahoe, USA, 1995, pp. 194–202.