Vol. 34 No. 7 Transactions of Information Processing Society of Japan July 1993
Regular Paper

Computing Soft Constraints by Hierarchical Constraint Logic Programming

KEN SATOH †☆ and AKIRA AIBA †

We have already formalized soft constraints based on interpretation ordering which is a generalization of circumscription. However, this formalization is written in a second-order formula and therefore is not computable in general. To make it computable, we have to introduce some restriction. In this paper, we propose a *semantic restriction.* By semantic restriction, we mean that we fix the considered domain so that interpretations of domain-dependent relations are fixed, and soft and hard constraints contain only domain-dependent relations. If we accept this restriction, the soft constraints can be expressed in a first-order formula. Moreover, there is already a proposed mechanism suitable for computing such restricted soft constraints in the literature, that is, hierarchical constraint logic programming languages (HCLP languages). Firstly, we identify a solution to constraint hierarchy defined by HCLP languages with the most preferable solution for semantically-restricted soft constraints. Then, we provide an algorithm for calculating all the most preferable solutions for soft constraints without any redundant calls a constraint solver. Then, we show examples of computing soft constraints by using our HCLP language named CHAL (Contraintes Hierarchiques avec Logique).

1. Introduction

In the area of *synthesis* problems such as job shop scheduling, circuit design and planning, there are two kinds of constraints. One kind is *hard constraints* in which every solution is required to satisfy and the other is *soft constraints* which provide preferences over solutions.4),11) Most systems manipulating soft constraints use evaluation functions to represent these soft constraints. However, it is hard to debug evaluation functions if obtained solutions are unsatisfactory. One possible solution to this problem is to manipulate soft constraints in a logical manner.

We have proposed a logical foundation of soft constraints in Ref. 10) by using a meta-language9) which expresses an interpretation ordering. The idea of formalizing soft constraints is as follows. Let hard constraints be represented in first-order formulas. Then an interpretation which satisfies all of these first-order formulas can be regarded as a possible solution and soft constraints can be regarded as an order over those interpretations because soft constraints represent criteria over possible solutions to choose the most preferable ones. We use a meta-language which represents a preference order directly. This meta-language can be translated into a second-order formula to provide a syntactical definition of the most preferable solutions.

Although this framework is rigorous and declarative, it is not computable in general because it is defined by a second-order formula. Therefore, we have to restrict a class of constraints so that these constraints are computable.

Since the above interpretation ordering is a generalization of circumscription, one might think that restrictions for computing circumscription can be helpful. However, previous proposals of restricting circumscription are domain-independent, that is, syntactical such as separable axioms5) for ordinary circumscription or stratified axioms6) for prioritized circumscription. Although these techniques are applicable in any domain, these restrictions are not so clear at identifying useful applications.

In this paper, we propose another restriction based on semantics. By *semantic restriction*, we mean that we fix the considered domain so that interpretations of domain-dependent relations are fixed, and soft and hard constraints consist of only domain-dependent relations. This restriction is fairly reasonable because when we solve any actual problem with soft constraints such as design and planning, we usually know the domain of the problem. If we accept this restriction, the soft constraints can be expressed in a first-order formula. Moreover, there is already a

†Institute for New Generation Computer Technology
☆Currently, Fujitsu Laboratories Ltd.

proposed mechanism suitable for computing such restricted soft constraints in the literature, that is, hierarchical constraint logic programming languages (HCLP languages).[2]

In the following section, we show a relationship between semantics of soft constraints in Ref. 10) and constraint hierarchy defined by HCLP language. By this relationship, we use HCLP to compute the most preferable solutions specified by semantically-restricted soft constraints.

Then, we show an algorithm to compute all simplified constraint sets of the most preferable solutions from constraint hierarchy without any redundant calls of a constraint solver and show examples of computing soft constraints by our HCLP language CHAL. Proofs of Theorems are found in Appendix B.

2. Constraint Hierarchy and Soft Constraints

In this section, we define HCLP and then, show a relationship between constraint hierarchy defined by HCLP and the soft constraints proposed in Ref. 10).

2.1 HCLP

We follow the definition of HCLP in Ref. 2) but extend it by introducing a complex form of soft constraints which allows disjunctions. HCLP language is a language augmenting CLP language with labeled constraints. An HCLP program consists of rules of the form:

$h{:}-b_1, \cdots, b_n$

where $h, b_1, \cdots, b_n$ are predicates or constraints or labeled constraints. A labeled constraint is of the form:

label C

where C is a complex constraint in which only domain-dependent functional symbols can be allowed as functional symbols and **label** is a label which expresses strength of the complex constraint C.

A complex constraint is a disjunction of conjunctions of domain-dependent constraints of the form:

$$((c_{11}, c_{12}, \cdots, c_{1n_1});\ (c_{21}, c_{22}, \cdots, c_{2n_2});\ \cdots;\ (c_{m1}, c_{m2}, \cdots, c_{mn_m}))$$

where $1 \leq m$ and $0 \leq n$, and ';' expresses a disjunction and ',' expresses a conjunction and c_{ij} is an atomic constraint whose constraint symbol is domain-dependent.

The operational semantics for HCLP is similar to CLP except manipulating constraint hierarchy. In HCLP, we accumulate labeled constraints to form constraint hierarchy by each label while executing CLP until CLP solves all goals and gives a reduced required constraints. Then, we solve constraint hierarchy with required constraints.

An assignment of variables is a mapping from each variable in constraints to an element of the considered domain. We say an assignment θ satisfies a constraint if the substitution of the free variables in the constraint makes the constraint true. An assignment of variables is partially ordered by the "better" comparator as follows.

Let θ and σ be assignments and C^1_θ (and C^1_σ) be a set of constraints in the strongest level of the hierarchy satisfied by θ (and σ), and C^2_θ (and C^2_σ) be a set of constraints in the second strongest level of the hierarchy satisfied by θ (and σ), $\cdots$, and C^k_θ (and C^k_σ) be a set of constraints in the k-th strongest level of the hierarchy satisfied by θ (and σ).

θ is *better* than σ w.r.t. the constraint hierarchy if there exists i $(1 \leq i \leq k)$ such that for every j $(1 \leq j \leq i-1)$, $g(C^j_\sigma) \cong g(C^j_\theta)$ and $g(C^i_\sigma) < g(C^i_\theta)$, where g expresses some values (possibly in the form of set) and $\cong$ expresses an equivalence relation over the value of g and $<$ expresses a strict partial order over the value of g.

If g returns the set itself and $\cong$ is an equivalence on sets and $<$ is a strict subset relation $\subset$, then we say that θ is *locally-predicate-better*[2] than σ w.r.t. the constraint hierarchy.

If g returns the number of elements in the set and $\cong$ is $=$ and $<$ is $<$, then we say that θ is *unsatisfied-count-better*[2] than σ w.r.t. the constraint hierarchy.

See another definitions of g and $\cong$ and $<$ in Ref. 2).

Then, a *solution w.r.t. the required constraints and the constraint hierarchy* is defined as an assignment θ which satisfies the required constraints and has no assignment σ such that σ satisfies the required constraints and σ is better than θ.

2.2 Relationship with Most Preferable Solutions

Now, we relate solutions w.r.t. required constraints and a constraint hierarchy with the most preferable solutions for soft constraints with priority defined in Ref. 10). We treat free variables in constraints as individual constants and regard an assignment for these variables as an interpretation and the order defined by "better" comparator as an order over these interpretations.

Here, we consider only two kind of orders; the orders defined by locally-predicate-better comparator and unsatisfied-count-better comparator. But, we can easily modify the results below for other comparators defined in Ref. 2).

2. 2. 1 Locally-Predicate-Better Comparison

Let A be the axioms of the considered domain D and $C_1^1(\mathbf{x}), \cdots, C_{m_1}^1(\mathbf{x})$ be constraints in the strongest level of the hierarchy and $C_1^2(\mathbf{x}), \cdots, C_{m_2}^2(\mathbf{x})$ be constraints in the second strongest level of the hierarchy, $\cdots$, and $C_1^k(\mathbf{x}), \cdots, C_{m_k}^k(\mathbf{x})$ be constraints in the k-th strongest level of the hierarchy where $\mathbf{x}$ is a tuple of all free variables in these constraints. We introduce a tuple of new constants, $\mathbf{X}$, by which we replace free variables so that we treat free variables as individual constants.

Let M' and M be interpretations with the domain D each of which satisfies A and differs from the other interpretation in at most the assignments of $\mathbf{X}$. Since M' and M differs in at most the assignments of individual constants, there is one-to-one correspondence between an interpretation and the above assignment of free variables. Now, we define an order over interpretations by the above constraint hierarchy in meta-language defined in Ref. 10) for locally-predicate-better comparator.

We write the relation expressing that M' is more preferable than M as $M' <_\phi^{lpb} M$[☆] and define the relation as follows:

$$M' <_\phi^{lpb} M \overset{def}{=} (M' \leq_\phi M) \mathbin{\mathbb{\wedge}} \neg (M \leq_\phi M')^{☆☆}$$

where $M' \leq_\phi M$ is an abbreviation of $(M' \leq_\phi^1 M) \mathbin{\mathbb{\wedge}} (M' \leq_\phi^2 M) \mathbin{\mathbb{\wedge}} \cdots \mathbin{\mathbb{\wedge}} (M' \leq_\phi^k M)$ and $M' \leq_\phi^i M$ is defined as follows:

$$(\mathbb{\wedge}_{j=1}^{i-1} \mathbb{\wedge}_{l=1}^{m_j} ((M \models_\phi C_l^j(\mathbf{X})) \equiv (M' \models_\phi C_l^j(\mathbf{X}))))$$
$$\supset (\mathbb{\wedge}_{l=1}^{m_i} ((M \models_\phi C_l^i(\mathbf{X})) \supset (M' \models_\phi C_l^i(\mathbf{X}))))$$

This relation intuitively means that interpretations which satisfy $C_1^1(\mathbf{X}), \cdots, C_{m_1}^1(\mathbf{X})$ as much as possible is preferable and if there are interpretations which satisfy the same constraints in the first place, then interpretations which satisfy $C_1^2(\mathbf{X}), \cdots, C_{m_2}^2(\mathbf{X})$ as much as possible are preferable and, $\cdots$ if there are interpretations which satisfy the same constraints up to the $(k-1)$-th place, then interpretations which satisfy $C_1^k(\mathbf{X}), \cdots, C_{m_k}^k(\mathbf{X})$ as much as possible are preferable.

Let A be the axioms of the considered domain. Then, the *most preferable solutions w.r.t. required constraints and the order* $<_\phi^{lpb}$ is a model M which satisfies A and the required constraints and there is no model M' such that M' satisfies A and the required constraints and $M' <_\phi^{lpb} M$. From the equivalence between an order defined by locally-predicate-better comparator and the order $<_\phi^{lpb}$, the most preferable solutions w.r.t. required constraints and the order $<_\phi^{lpb}$ are equivalent to the solutions w.r.t. required constraints and constraint hierarchy for a locally-predicate-better comparator.

We can give a syntactic definition of the most preferable solutions from the result in Ref. 10). Let A be the axioms of the considered domain and $RC(\mathbf{X})$ be a conjunction of required constraints where $\mathbf{X}$ be a tuple of the newly introduced constants and $\mathbf{x}$ be a tuple of variables.

$$A \wedge RC(\mathbf{X}) \wedge$$
$$\neg \exists \mathbf{x} (RC(\mathbf{x}) \wedge (\mathbf{x} \leq \mathbf{X}) \wedge \neg (\mathbf{X} \leq \mathbf{x})) \qquad (\mathrm{P}_{lpb})$$

where $\mathbf{x} \leq \mathbf{X}$ is an abbreviation of $(\mathbf{x} \leq^1 \mathbf{X}) \wedge \cdots \wedge (\mathbf{x} \leq^k \mathbf{X})$ **and** $\mathbf{x} \leq^i \mathbf{X}$ is an abbreviation of the following formula:

$$\left(\bigwedge_{j=1}^{i-1} \bigwedge_{l=1}^{m_j} (C_l^j(\mathbf{X}) \equiv C_l^j(\mathbf{x})) \right) \supset$$
$$\left(\bigwedge_{l=1}^{m_i} (C_l^i(\mathbf{X}) \supset C_l^i(\mathbf{x})) \right).$$

Adapted from the result in Ref. 10), we can show the following theorem.

Theorem 1 *M is a most preferable solution w.r.t. $RC(\mathbf{X})$ and the order $<_\phi^{lpb}$ if and only if M is a model of the formula (P_{lpb}).*

Example 1 *Let A be axioms for addition and multiplication of integers and $RC(x, y) = (x * y = 8)$, $C_1^1(x, y) = (x = 4)$ and $C_1^2(x, y) = (y = 4)$.*

This means that $x = 4$ and $y = 4$ should be satisfied as much as possible but there is a

---

☆ ϕ is an assignment function for free variables used for satisfaction relation $\models_\phi$.

☆☆ Hereafter, we use $\mathbb{\wedge}$, $\supset$, $\equiv$ and $\neg$ as meta-logical connectives for "conjunction," "implication," "equivalence" and "negation."

priority of $x=4$ to $y=4$. In this case, since $RC(x, y)$ is $x * y=8$, $x=4$ can be satisfied but $y=4$ will be ignored. We show that this result corresponds with the result from the formula (P_{lpb}).

(P_{lpb}) becomes[*]:

$$A \wedge (X * Y=8) \wedge \neg \exists a \exists b \\ ((a * b=8) \wedge \\ ((X=4) \supset (a=4)) \wedge (((X=4) \equiv (a=4)) \supset ((Y=4) \supset (b=4))) \wedge \\ \neg (((a=4) \supset (X=4)) \wedge (((a=4) \equiv (X=4)) \supset ((b=4) \supset (Y=4)))))$$

which is equivalent to:

$$A \wedge (X * Y=8) \wedge \forall a \forall b \\ \neg ((a * b=8) \wedge \\ ((X=4) \supset (a=4)) \wedge (((X=4) \equiv (a=4)) \supset ((Y=4) \supset (b=4))) \wedge \\ \neg (((a=4) \equiv (X=4)) \wedge ((b=4) \equiv (Y=4)))).$$

The above formula must always be true for every assignment for a and b. Suppose we assign 4 to a and 2 to b. Then, the above formula is equivalent to:

$$A \wedge (X * Y=8) \wedge \neg ((4 * 2=8) \wedge \\ ((X=4) \supset (4=4)) \wedge (((X=4) \equiv (4=4)) \supset ((Y=4) \supset (2=4))) \wedge \\ \neg (((4=4) \equiv (X=4)) \wedge ((2=4) \equiv (Y=4))))$$

which is equivalent to:

$$A \wedge (X * Y=8) \wedge \neg (((X=4) \supset (Y \neq 4)) \wedge \neg (X=4 \wedge Y \neq 4))$$

which is reduced to:

$$A \wedge (X=4) \wedge (Y=2)$$

which corresponds with the result from the constraint hierarchy. □

Although the general definition of the most preferable model is written in a second-order formula, the above formula is a first-order formula, that is, computable. It is because each of the above constraints is a logical combination of domain-dependent constraints with a fixed interpretation and therefore, only parameters in the above formula are individual constants in constraints.

From the point of view of soft constraints, we can regard this restriction as a *semantic restriction* because we fix the considered domain and use only domain-dependent constraints. This restriction is fairly reasonable because when we solve actual problems with soft constraints such as design and planning, we usually know the domain of the problem.

[*]We replace free variables (x, y) in constraints by new individual constants (X, Y).

Now, we show a relationship between the most preferable models and maximal consistent sets w.r.t. required constraints and constraint hierarchy for a locally-predicate-better comparator. This is the key relation to calculate the most preferable model in HCLP languages.

Firstly, we define *maximal consistent set of constraints*.

Definition 1 *Let A be the axioms of the considered domain and $RC(\mathbf{X})$ be a conjunction of required constraints and $CH(\mathbf{X})$ be a constraint hierarchy with k levels. A set of constraints $MC(\mathbf{X})$ is maximal consistent w.r.t. A and $RC(\mathbf{X})$ and $CH(\mathbf{X})$ for locally-predicate-better comparator if $MC(\mathbf{X})$ satisfies the following conditions.*

1. *$MC(\mathbf{X})$ is consistent with A.*
2. *$MC(\mathbf{X})$ is logically equivalent to $RC(\mathbf{X}) \wedge CH^1(\mathbf{X}) \wedge \cdots \wedge CH^k(\mathbf{X})$ where $CH^i(\mathbf{X})$ is a conjunction of some constraints in the i-th level of $CH(\mathbf{X})$.*
3. *There is no consistent set of constraints $MC'(\mathbf{X})$ with A which is logically equivalent to $RC(\mathbf{X}) \wedge CH'^1(\mathbf{X}) \wedge \cdots \wedge CH^k(\mathbf{X})$ where $CH'^i(\mathbf{X})$ is a conjunction of some constraints in the i-th level of $CH(\mathbf{X})$ and satisfies the following condition:*
   there exists i $(1 \leq i \leq k)$ such that for every j $(1 \leq j \leq i-1)$, $CH^j(\mathbf{X}) = CH'^j(\mathbf{X})$ and
   $$CH^i(\mathbf{X}) \subset CH'^i(\mathbf{X})$$

Then, a relation between maximal consistent sets and the most preferable models is as follows.

Theorem 2 *Let A be the axioms of the considered domain and $RC(\mathbf{X})$ be a conjunction of required constraints and $CH(\mathbf{X})$ be a constraint hierarchy. Let $MC_1(\mathbf{X}), MC_2(\mathbf{X}), \cdots, MC_n(\mathbf{X})$ be all maximally consistent sets (which are logically different from each other) w.r.t. A and $RC(\mathbf{X})$ and $CH(\mathbf{X})$ for locally-predicate-better comparator.*

Then, $A \wedge (MC_1(\mathbf{X}) \vee MC_2(\mathbf{X}) \vee \cdots \vee MC_n(\mathbf{X}))$ is logically equivalent to the formula (P_{lpb}). In other words, models of $A \wedge (MC_1(\mathbf{X}) \vee MC_2(\mathbf{X}) \vee \cdots \vee MC_n(\mathbf{X}))$ are exactly all the most preferable models.

Example 2 *Consider the constraint hierarchy* *of Example 1. Then, the maximal consistent set for the constraint hierarchy is only $X=4\wedge Y=2$ which corresponds with the result from the formula (P_{lpb}).* □

If we have a *satisfaction-complete* constraint solver which can determine whether a set of constraints is consistent or not, then we neither need to write the first-order axioms of the domain nor need to use first-order inference rules to infer the most preferable models. All we have to do is to write constraints directly in HCLP and use the satisfaction-complete solver to compute all maximal consistent sets by generating every consistent subset of the constraint hierarchy and checking if it is maximal.

2.2.2 Unsatisfied-Count-Better Comparison

In this subsection, we show that we can extend the above results to unsatisfied-count-better comparators. Actually, however, we can extend our results to other comparators defined in Ref. 2) in a similar way to the below.

Let A be the axioms of the considered domain D and $C_1^1(\mathbf{x}), \cdots, C_{m_1}^1(\mathbf{x})$ be constraints in the strongest level of the hierarchy and $C_1^2(\mathbf{x}), \cdots, C_{m_2}^2(\mathbf{x})$ be constraints in the second strongest level of the hierarchy, $\cdots$, and $C_1^k(\mathbf{x}), \cdots, C_{m_k}^k(\mathbf{x})$ be constraints in the k-th strongest level of the hierarchy where $\mathbf{x}$ is a tuple of all free variables in these constraints. Same as above, we introduce a tuple of new constants by which we replace free variables so that we treat free variables as individual constants.

We introduce the following axioms in order to count the number of unsatisfied constraints. For every constraint $C(\mathbf{X})$, we define $C(\mathbf{X})\equiv(E_C=0)$ and $\neg C(\mathbf{X})\equiv(E_C=1)$ where E_C corresponds with the trivial error function[2)] for the constraint $C(\mathbf{X})$. We also assume axioms for integers. Let $\mathbf{E}$ be a tuple of the above E_Cs and $B(\mathbf{X}, \mathbf{E})$ be the above axiom set.

Let M' and M be interpretations with the domain D each of which satisfies A and $B(\mathbf{X}, \mathbf{E})$, and differs from the other interpretation in at most the assignments of $\mathbf{X}$ and $\mathbf{E}$. Let Φ_D be a set of all assignment function for variables and ϕ_x be an assignment function which differs from ϕ in at most the assignment of x and ϕ_{xy} be an assignment function which differs from ϕ_x in at most the assignment of y*. Now, we define an order over interpretations by the above constraint hierarchy in meta-language defined in Ref. 10) for unsatisfied-count-better comparator. We write the relation expressing that M' is more preferable than M as $M'<_{\phi}^{ucb}M$:

$$M'<_{\phi}^{ucb}M \overset{def}{=} (M'\leq M)\mathbin{\wedge\!\!\!\wedge}\neg(M\leq M'),$$

where $M'\leq M$ is an abbreviation of $(M'\leq^1 M)\mathbin{\wedge\!\!\!\wedge}(M'\leq^2 M)\mathbin{\wedge\!\!\!\wedge}\cdots\mathbin{\wedge\!\!\!\wedge}(M'\leq^k M)$ and $M'\leq^i M$ is defined as follows:

$$\begin{aligned}&(\textstyle\bigwedge_{j=1}^{i-1}(\forall\phi_x\in\Phi_D\forall\phi_{xy}\in\Phi_D(\\&\quad((M'\models_{\phi_{xy}}(x=\textstyle\sum_{l=1}^{m_j}E_l^j))\mathbin{\wedge\!\!\!\wedge}(M\models_{\phi_{xy}}(y\\&\qquad=\textstyle\sum_{l=1}^{m_j}E_l^j)))\supset(M'\models_{\phi_{xy}}x=y))))\supset\\&(\forall\phi_x\in\Phi_D\forall\phi_{xy}\in\Phi_D(\\&\quad((M'\models_{\phi_{xy}}(x=\textstyle\sum_{l=1}^{m_i}E_l^i))\mathbin{\wedge\!\!\!\wedge}(M\models_{\phi_{xy}}(y\\&\qquad=\textstyle\sum_{l=1}^{m_i}E_l^i)))\supset(M'\models_{\phi_{xy}}x\leq y)))\end{aligned}$$

where E_l^i is the trivial error function for the constraint $C_l^i(\mathbf{X})$.

This relation intuitively means that interpretations which satisfy $C_1^1(\mathbf{X}), \cdots, C_{m_1}^1(\mathbf{X})$ as many as possible is preferable** and if there are interpretations which satisfy the same number of constraints in the first place, then interpretations which satisfy $C_1^2(\mathbf{X}), \cdots, C_{m_2}^2(\mathbf{X})$ as many as possible are preferable and, $\cdots$ if there are interpretations which satisfy the same number of constraints up to the $(k-1)$-th place, then interpretations which satisfy $C_1^k(\mathbf{X}), \cdots, C_{m_k}^k(\mathbf{X})$ as many as possible are preferable.

Let A be the axioms of the considered domain and $B(\mathbf{X}, \mathbf{E})$ be the axiom for the error function. Then, the *most preferable solutions w.r.t. required constraints and the order* $<_{\phi}^{ucb}$ is a model M which satisfies A and $B(\mathbf{X}, \mathbf{E})$ and the required constraints, and there is no model M' such that M' satisfies A and $B(\mathbf{X}, \mathbf{E})$ and the required constraints and $M'<_{\phi}^{ucb}M$. From the equivalence between an order defined by unsatisfied-count-better comparator and the order $<_{\phi}^{ucb}$, the most preferable solutions w.r.t. required constraints and the order $<_{\phi}^{ucb}$ are equivalent to the solutions w.r.t. required constraints and constraint hierarchy.

We can give a syntactic definition of the most preferable solutions in a similar way to Section 2.2.1. Let A be the axioms of the considered domain and $B(\mathbf{X}, \mathbf{E})$ be the axiom set for the error function and $RC(\mathbf{X})$ be a conjunction of required constraints where $\mathbf{X}$ be a tuple of the newly introduced constants and $\mathbf{x}$ and $\mathbf{e}$ be

*A precise definition can be found in Ref. 10).

**This means that the number of unsatisfied constraints is minimum.

tuples of variables.

$$A \wedge B(\mathbf{X}, \mathbf{E}) \wedge RC(\mathbf{X}) \wedge \neg \exists \mathbf{x} \exists \mathbf{e} (B(\mathbf{x}, \mathbf{e}) \wedge RC(\mathbf{x}) \wedge (\langle \mathbf{x}, \mathbf{e} \rangle \leq \langle \mathbf{X}, \mathbf{E} \rangle) \wedge \neg (\langle \mathbf{X}, \mathbf{E} \rangle \leq \langle \mathbf{x}, \mathbf{e} \rangle)) \qquad (\mathrm{P}_{ucb})$$

where $\langle \mathbf{x}, \mathbf{e} \rangle \leq \langle \mathbf{X}, \mathbf{E} \rangle$ is an abbreviation of:

$$(\langle \mathbf{x}, \mathbf{e} \rangle \leq^1 \langle \mathbf{X}, \mathbf{E} \rangle) \wedge \cdots \wedge (\langle \mathbf{x}, \mathbf{e} \rangle \leq^k \langle \mathbf{X}, \mathbf{E} \rangle)$$

and $\langle \mathbf{x}, \mathbf{e} \rangle \leq^i \langle \mathbf{X}, \mathbf{E} \rangle$ is an abbreviation of the following formula:

$$\left(\bigwedge_{j=1}^{i-1} \left(\sum_{l=1}^{m_j} e_l^j = \sum_{l=1}^{m_j} E_l^j \right) \right) \supset \left(\sum_{l=1}^{m_i} e_l^i \leq \sum_{l=1}^{m_i} E_l^i \right).$$

We can show a similar theorem to Theorem 1.

Theorem 3 *M is a most preferable solution w.r.t. $RC(\mathbf{X})$ and the order $<_{\phi}^{ucb}$ if and only if M is a model of the formula (P_{ucb}).*

Example 3 *Let A be axioms for addition and multiplication of integers and $RC(x, y) = (x * y = 8)$, $C_1^1(x, y) = (x=4)$, $C_2^1(x, y) = (y=2)$ and $C_3^1(x, y) = (x=2)$.*

This means that $x=4$ and $y=2$ and $x=2$ should be satisfied as many as possible. In this case, since $RC(x, y)$ is $x * y=8$, $\langle x=4, y=2 \rangle$ is maximum in the number of consistent combinations of constraints. We show that this result corresponds with the result from the formula (P_{ucb}).

Firstly, we assume the following axioms for the error function[*].

$$((X=4) \equiv (E_1^1=0)) \wedge ((X \neq 4) \equiv (E_1^1=1)) \wedge ((Y=2) \equiv (E_2^1=0)) \wedge ((Y \neq 2) \equiv (E_2^1=1)) \wedge ((X=2) \equiv (E_3^1=0)) \wedge ((X \neq 2) \equiv (E_3^1=1))$$

We denote the above set as $B(X, Y, E_1^1, E_2^1, E_3^1)$.

Then, (P_{ucb}) becomes:

$$A \wedge B(X, Y, E_1^1, E_2^1, E_3^1) \wedge (X * Y=8) \wedge \neg \exists a \exists b \exists e_1 \exists e_2 \exists e_3 (B(x, y, e_1, e_2, e_3) \wedge (a * b=8) \wedge ((e_1+e_2+e_3) \leq (E_1^1+E_2^1+E_3^1)) \wedge \neg ((E_1^1+E_2^1+E_3^1) \leq (e_1+e_2+e_3)))$$

which is equivalent to:

$$A \wedge (X * Y=8) \wedge \forall a \forall b \forall e_1 \forall e_2 \forall e_3 \neg (B(x, y, e_1, e_2, e_3) \wedge (a * b=8) \wedge ((e_1+e_2+e_3) < (E_1^1+E_2^1+E_3^1)))$$

The above formula must always be true for every assignment for a and b. Suppose we assign 4 to a, 2 to b, 0 to e_1, 0 to e_2 and 1 to e_3. Then, the above formula is equivalent to:

[*]We replace free variables (x, y) in constraints by new individual constants (X, Y).

$$A \wedge B(X, Y, E_1^1, E_2^1, E_3^1) \wedge (X * Y=8) \wedge \neg (B(4, 2, 0, 0, 1) \wedge (4 * 2=8) \wedge ((0+0+1) < (E_1^1+E_2^1+E_3^1)))$$

which is equivalent to:

$$A \wedge B(X, Y, E_1^1, E_2^1, E_3^1) \wedge (X * Y=8) \wedge ((E_1^1+E_2^1+E_3^1) \leq 1).$$

This means that the number of unsatisfied constraints must be 0 or 1. However, from $B(X, Y, E_1^1, E_2^1, E_3^1)$, the former case is impossible. Therefore, $B(X, Y, E_1^1, E_2^1, E_3^1)$ becomes:

$$((X \neq 4) \wedge (Y=2) \wedge (X=2)) \vee ((X=4) \wedge (Y \neq 2) \wedge (X=2)) \vee ((X=4) \wedge (Y=2) \wedge (X \neq 2))$$

which is equivalent to $X=4 \wedge Y=2$ from the required constraints. This corresponds with the result from the constraint hierarchy with unsatisfied-count-better comparator. □

For unsatisfied-count-better comparator, we can also define a similar notion of maximal consistent set and show a similar correspondence to Section 2.2.1.

Definition 2 *Let A be the axioms of the considered domain and $B(\mathbf{X}, \mathbf{E})$ be an axiom set for the error function and $RC(\mathbf{X})$ be a conjunction of required constraints and $CH(\mathbf{X})$ be a constraint hierarchy with k levels. A set of constraints $MC(\mathbf{X})$ is maximal consistent w.r.t. A and $B(\mathbf{X}, \mathbf{E})$ and $RC(\mathbf{X})$ and $CH(\mathbf{X})$ for unsatisfied-count-better comparator if $MC(\mathbf{X})$ satisfies the following conditions.*

1. *$MC(\mathbf{X})$ is consistent with A and $B(\mathbf{X}, \mathbf{E})$.*
2. *$MC(\mathbf{X})$ is logically equivalent to $RC(\mathbf{X}) \wedge CH^1(\mathbf{X}) \wedge \cdots \wedge CH^k(\mathbf{X})$ where $CH^i(\mathbf{X})$ is a conjunction of some constraints in the i-th level to $CH(\mathbf{X})$.*
3. *There is no consistent set of constraints $MC'(\mathbf{X})$ with A and $B(\mathbf{X}, \mathbf{E})$ which is logically equivalent to $RC(\mathbf{X}) \wedge CH'^1(\mathbf{X}) \wedge \cdots \wedge CH'^k(\mathbf{X})$ where $CH'^i(\mathbf{X})$ is a conjunction of some constraints in the i-th level of $CH(\mathbf{X})$ and satisfies the following condition:*

   there exists i $(1 \leq i \leq k)$ such that
   for every j $(1 \leq j \leq i-1)$,
   $|CH^j(\mathbf{X})| = |CH'^j(\mathbf{X})|$ and
   $|CH^i(\mathbf{X})| < |CH'^i(\mathbf{X})|$

   where $|CH^j(\mathbf{X})|$ is the number of constraints in $CH^j(\mathbf{X})$.

Theorem 4 *Let A be the axioms of the considered domain and $B(\mathbf{X}, \mathbf{E})$ be an axiom set for the error function and $RC(\mathbf{X})$ be a con-*

junction of required constraints and $CH(\mathbf{X})$ be a constraint hierarchy. Let $MC_1(\mathbf{X}), MC_2(\mathbf{X}), \cdots, MC_n(\mathbf{X})$ be all maximal consistent sets (which are logically different from each other) w.r.t. A and $B(\mathbf{X}, \mathbf{E})$ and $RC(\mathbf{X})$ and $CH(\mathbf{X})$ for unsatisfied-count-better comparator. Then, $A \wedge B(\mathbf{X}, \mathbf{E}) \wedge (MC_1(\mathbf{X}) \vee MC_2(\mathbf{X}) \vee \cdots \vee MC_n(\mathbf{X}))$ is logically equivalent to the formula (P_{ucb}). In other words, models of $A \wedge B(\mathbf{X}, \mathbf{E}) \wedge (MC_1(\mathbf{X}) \vee MC_2(\mathbf{X}) \vee \cdots \vee MC_n(\mathbf{X}))$ are exactly all the most preferable models.

Example 4 *Consider the constraint hierarchy of Example 3. Then, the maximal consistent set w.r.t. the constraint hierarchy for unsatisfied-count-better comparator is only $X=4 \wedge Y=2$ which corresponds with the result from the formula (P_{ucb}).* □

Same as Section 2.2.1, we can use a satisfaction-complete constraint solver to compute all maximal consistent sets by generating every consistent subset of the constraint hierarchy and checking if it is maximal in terms of unsatisfied-count-better comparator.

3. Algorithm to Solve Constraint Hierarchy

We show an algorithm for solving constraint hierarchy in Appendix A. Inputs of the algorithm are a constraint hierarchy and a set of reduced required constraints and its output is all maximal consistent sets of constraints simplified by the constraint solver. Features of this algorithm are as follows.

1. There is no redundant calls of constraint solver for the same combination of constraints since it calculates reduced constraints in bottom-up manner.
2. If an inconsistent combination of constraints is found by calling constraint solver, it is registered as a nogood and used for further contradiction detecting and any extension of the combination will not be processed to avoid vein combinations.
3. Inconsistency is detected without a call of constraint solver if a processed combination subsumes a registered nogood.

In this algorithm, a quadruple ⟨*Cs*, *UC*, *RC*, *Rest*⟩ plays a center role as follows.

1. *Cs*: A set of constraints which have been combined already; This set is used to check maximality of constraints. If a constraint is disjunctive, the disjunctive constraint itself is stored in this set.
2. *UC*: A set of constraints which have been used already; This set is used to check whether it is subsumed by any nogood. If this set is found to be inconsistent, the set is registered as a nogood. Note that *UC* is different from *Cs* in manipulation of disjunctive constraints. If a constraint is disjunctive, a disjunct actually used in the disjunctive constraint is stored in this set.
3. *RC*: Reduced constraint set from *UC* by a constraint solver; This set is used for output. Although this set can be used for nogood check by using the constraint solver, a computational cost is more expensive than the cost to use *UC*. Therefore, we do not use *RC* but *UC* for nogood check.
4. *Rest*: A set of constraints which has not been used yet; This set is necessary to avoid redundant check for consistency. Moreover, if a combined combination is found to be inconsistent, we no longer add constraints in *Rest* to the inconsistent combination so that a vein combination can be avoided.

The execution example of the bottom-up algorithm is shown in Fig. 1. In Fig. 1, we only consider the one level of hierarchy in which there are five constraints *P*, *Q*, *R*, *S* and *T* and we assume that combinations ⟨*P*, *Q*⟩ and ⟨*Q*, *R*⟩ are contradictory. The execution proceeds from the bottom to the top and from the left to the right. We begin with the bottom circle where we consider only hard constraints. Then, we obtain five sets of constraints (the second line from the bottom) by adding each constraints and checking each consistency. The lines between circles express dependency of combination of constraints which corresponds with a constraint added from *Rest*. Then, we proceed to the third line by adding more constraints. In the third line, the crossed circles express inconsistent combinations. If we find inconsistent combination, we no longer add constraints in *Rest* to the inconsistent combination and we register the combination (stored as *UC*) so that a vein consistency check can be avoided. In this

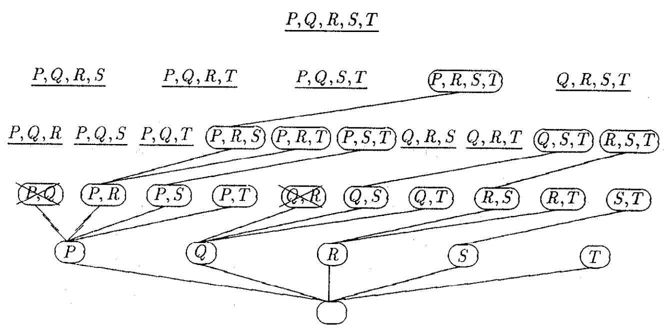


Fig. 1 Bottom-up algorithm.

example, under-lined sets of constraints are not checked. After having all consistent sets, we check if each of them is maximal by comparing each *Cs*. If we use locally-predicate-better comparator, ⟨*P*, *R*, *S*, *T*⟩ and ⟨*Q*, *S*, *T*⟩ are solutions and if we use unsatisfied-count-better comparator, ⟨*P*, *R*, *S*, *T*⟩ is the solution.

Borning et al.[2] have proposed the algorithm for constraint hierarchy. Our algorithm is different from theirs in the following points.

1. Their algorithm does not consider disjunctions of constraints in constraint hierarchy whereas ours can handle them.
2. Their algorithm consider only locally-predicate-better comparator whereas ours can be applied to unsatisfied-count-better comparator as well.
3. Their algorithm uses backtrack to get an alternative solution and so, it may call the constraint solver for the same combination of constraints redundantly.

Although our algorithm has no redundant calls of a constraint solver, it will call the constraint solver 2^n-1 times in the worst case where n is a number of soft constraints. So, we must ensure that inconsistency occurs at a small combination of constraints or we must prioritize constraints almost linearly. If we linearize constraints completely, then our algorithm will call constraint solver only n times.

Our algorithm has been implemented already on PSI (Personal Sequential Inference) Machine developed in ICOT. By using the algorithm, we have implemented an HCLP language called CHAL (Contraintes Hierarchiques avec Logique), an extension of CAL (Contraintes avec Logique)[7] which is a CLP language developed in ICOT. In CHAL, we can use the following constraint solvers in CAL. One is an algebraic constraint solver which manipulates multivariate polynomial equations based on Buchberger algorithm[3] to calculate Gröbner bases and the other is a Boolean constraint solver which extends Buchberger algorithm to handle propositional Boolean equations[8].

4. Examples

Now, we show two CHAL examples of calculating the most preferable solutions. One is a meeting scheduling problem solved by Boolean CHAL and the other is a multi-axis gearbox design problem solved by algebraic CHAL.

4.1 Meeting Scheduling Problem

In this subsection, we show an example of solving meeting scheduling problem in Ref. 10) by using Boolean CHAL.

In a Boolean CHAL program, we express constraints as Boolean equations and in Boolean equations, we can use the only constraint symbol, = and the function symbols such as /\ (conjunction), \/ (disjunction), -> (implication), <-> (equivalence), ~ (negation) and the constants such as 1 as truth and 0 as falsity and propositional variables. Note that /\, \/, ->, <->, ~ are function symbols in this representation since we consider the domain of Boolean value and we use the above symbols as Boolean functions.

Firstly, we regard the following terms as propositional variables. $c(x)$ represents that the meeting will be held on day x and $p(x)$, $v(x)$, $m(x)$ represent that the president, the vice president and the manager attend the meeting on day x.

Suppose that we consider a meeting schedule for Monday, Tuesday and Wednesday, and the

```
meeting1 :-
    bool:(c(mon)\/c(tue)\/c(wed))=1,
    hard([mon,tue,wed]),soft([mon,tue,wed]),
    bool:p(mon)=0,bool:m(tue)=0.
meeting2 :- meeting1,bool:v(wed)=0.
hard([]).
hard([X|Y]):-
    bool:(c(X)<->p(X))=1,bool:(v(X)->c(X))=1,bool:(m(X)->c(X))=1,
    hard(Y).
soft([]).
soft([X|Y]):-
    chal:soft(bool:(c(X)->v(X))=1,0),chal:soft(bool:(c(X)->m(X))=1,1),
    soft(Y).
```

Fig. 2 CHAL program for meeting scheduling.

following hard constraints represented in Boolean equations exist.

1. The meeting must be held:
   $(c(mon)\backslash/c(tue)\backslash/c(wed))=1$.
2. The president must attend the meeting:
   $(c(x)\text{-}\rangle p(x))=1$ for all $x=mon$, tue, wed.
3. Since $p(x)$ expresses that the president attends the meeting on day x, if it is true, $c(x)$ (the meeting is held on day x) is also true:
   $(p(x)\text{-}\rangle c(x))=1$ for all $x=mon$, tue, wed.
4. The same thing holds if the vice president or the manager attends the meeting. We can expand these constraints as follows:
   $(v(x)\text{-}\rangle c(x))=1$ for all $x=mon$, tue, wed,
   $(m(x)\text{-}\rangle c(x))=1$ for all $x=mon$, tue, wed,
5. The president cannot attend the meeting on Monday and the manager cannot attend the meeting on Tuesday:
   $p(mon)=0$ and $m(tue)=0$

And we consider the following soft constraints.

1. The vice president should *preferably* attend the meeting. This soft constraint means that $(c(x)\text{-}\rangle v(x))=1$ should be satisfied as much as possible for all $x=$ *mon*, *tue*, *wed*.
2. The manager also should *preferably* attend the meeting. This soft constraint means that $(c(x)\text{-}\rangle m(x))=1$ should be satisfied as much as possible for all $x=$ *mon*, *tue*, *wed*
3. The schedule of the vice president is *prioritized to the schedule of the manager. This priority means that* $(c(x)\text{-}\rangle v(x))=1$ is stronger than $(c(y)\text{-}\rangle m(y))=1$ for every $x=mon$, tue, wed and $y=$ *mon*, *tue*, *wed*. To do so, we attach the stronger label to $(c(x)\text{-}\rangle v(x))=1$ than to $(c(y)\text{-}\rangle m(y))=1$ for every $x=mon$, tue, wed and $y=mon$, tue, wed.

In this case, the order over solutions is defined by the locally-predicate-better comparator.

Then, a Boolean CHAL program which builds constraint hierarchy of the above example is shown in Fig. 2. In Fig. 2,

chal: soft(bool: (c(X)->v(X))=1, 0)

and

chal: soft(bool: (c(X)->m(X))=1, 1)

express soft constraints. The first argument is a constraint and the second argument expresses the strength of the constraint. In CHAL, we use a natural number to express the strength. A soft constraint with the number 0 is the strongest and a constraint becomes weaker as the associated number becomes bigger.

If we ask ?-meeting1, then Boolean CHAL firstly calculates a set of reduced constraints from required constraints and then computes all simplified maximal consistent sets of constraints by solving constraint hierarchy. In this case, Boolean CHAL returns only one maximal consistent set which includes $c(mon)=0$, $c(tue)=0$, $c(wed)=1$ (Fig. 3). Since on Wednesday, all of three can attend the meeting, Wednesday is selected for the most preferable date for the meeting.

The conclusion may be withdrawn by adding another constraint. For example, suppose a new constraint that the vice president cannot attend the meeting on Wednesday is added. That is, the following constraint is added:

$v(wed)=0$

This is done by asking ?-meeting2, and Boolean CHAL returns only one maximal consistent set which includes $c(mon)=0$, $c(tue)=1$, $c(wed)=0$ (Fig. 4). This means that Tues-

```
?- meeting1.
After considering soft constraints
solution
m(mon) = 0 .
m(tue) = 0 .
m(wed) = 1 .
p(mon) = 0 .
p(tue) = 0 .
p(wed) = 1 .
c(mon) = 0 .
c(tue) = 0 .
c(wed) = 1 .
v(mon) = 0 .
v(tue) = 0 .
v(wed) = 1 .

solutionend
```

Fig. 3 The solution to ?-meeting1.

```
?- meeting2.
After considering soft constraints
solution
m(mon) = 0 .
m(tue) = 0 .
m(wed) = 0 .
p(mon) = 0 .
p(tue) = 1 .
p(wed) = 0 .
c(mon) = 0 .
c(tue) = 1 .
c(wed) = 0 .
v(mon) = 0 .
v(tue) = 1 .
v(wed) = 0 .

solutionend
```

Fig. 4 The solution to ?-meeting2.

day is the most preferable meeting date in this new situation because the schedule of the vice president has the priority to the schedule of the manager. This expresses nonmonotonic character of soft constraints.

4.2 Gear Design

In this subsection, we show an example in algebraic CHAL adapted from a design problem for a multi-axis gearbox. A gearbox is used to produce various speeds from the main spin. Figure 5 shows an example of a three-axis gearbox.

Gears on *Axis*1 (g_1, g_2 in Fig. 5) and *Axis*3 (g_7, g_8 in Fig. 5) are fixed and gears on *Axis*2 (g_3, g_4, g_5, g_6 in Fig. 5) are slidable. Slidable gears slide along the axis and mesh with fixed gears. In Fig. 5, each pair of $\langle g_1, g_3\rangle$, $\langle g_2, g_4\rangle$, $\langle g_5, g_7\rangle$, $\langle g_6, g_8\rangle$, can mesh. Since there are two gear changes between *Axis*1 and *Axis*2, and two gear changes between *Axis*2 and *Axis*3, there are

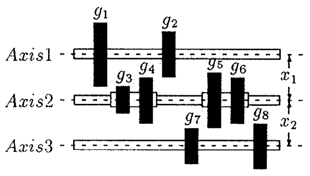


Fig. 5 Multi-Axis gearbox.

totally four output speeds ($=2 * 2$) by combinations of those gear changes.

Here, we consider the following design problem for a three-axis gearbox. We specify the sum of the two intervals between axes and output speed ratios each of which is produced by the combinations of two meshing pairs. We assume that there are standard radii of gears which take discrete values. We calculate each radius of gears so that standard gears are used as many as possible.

For example, suppose that we give the following specification for a four-speed gearbox shown in Fig. 5. We denote a speed ratio by a combination of two meshing pairs $\langle g_i, g_j\rangle$ and $\langle g_k, g_l\rangle$ as $ratio(\langle g_i, g_j\rangle, \langle g_k, g_l\rangle)$. Then, we specify ratios as follows.

$$ratio(\langle g_1, g_3\rangle, \langle g_5, g_7\rangle)=1 \tag{a}$$
$$ratio(\langle g_2, g_4\rangle, \langle g_5, g_7\rangle)=2 \tag{b}$$
$$ratio(\langle g_1, g_3\rangle, \langle g_6, g_8\rangle)=4 \tag{c}$$
$$ratio(\langle g_2, g_4\rangle, \langle g_6, g_8\rangle)=8 \tag{d}$$

And we specify the sum of intervals as 10 and standard values of radius as 1, 2, 3, 4.

Now, we modify this specification into a set of constraints expressed in equations. Let r_i be the radius of the gear g_i. Since $ratio(\langle g_i, g_j\rangle, \langle g_k, g_l\rangle)=p$ can be translated into an equation $r_i * r_k = r_j * r_l * p$, (a), $\cdots$, (d) are translated into the following equations.

$$r_1 * r_5 = 1 * r_3 * r_7 \tag{1}$$
$$r_2 * r_5 = 2 * r_4 * r_7 \tag{2}$$
$$r_1 * r_6 = 4 * r_3 * r_8 \tag{3}$$
$$r_2 * r_6 = 8 * r_4 * r_8 \tag{4}$$

Note that one of the above four equations is redundant.

From the condition of meshing, the sum of radii between meshing pair must be equal to the interval between axes. We denote the interval between *Axis*1 and *Axis*2 as x_1 and the interval between *Axis*2 and *Axis*3 as x_2. Then, the following constraints exist.

$$r_1 + r_3 = x_1 \tag{5}$$
$$r_2 + r_4 = x_1 \tag{6}$$

```
gear :-
     ratio(1,2,4,8),distance(10),
     pos_val([r1,r2,r3,r4,r5,r6,r7,r8],[1,2,3,4]).
ratio(P1,P2,P3,P4) :-
    alg:r1*r5=P1*r3*r7, alg:r2*r5=P2*r4*r7,
    alg:r1*r6=P3*r3*r8, alg:r2*r6=P4*r4*r8.
distance(D) :-
    alg:x1+x2=D,
    alg:r1+r3=x1,alg:r2+r4=x1,
    alg:r5+r7=x2,alg:r6+r8=x2.
pos_val([],_).
pos_val([R|RL],VL) :- pos_val1(R,VL,C),chal:soft(C,0),pos_val(RL,VL).
pos_val1(R,[X],alg:R=X).
pos_val1(R,[X|Y],(alg:R=X;C)):- pos_val1(R,Y,C).
```

Fig. 6 CHAL program for gearbox design.

```
?- gear.
After considering soft constraints
solution
r1 = 3 .
r2 = 4 .
r3 = 3 .
r4 = 2 .
r5 = 2 .
r6 = 16/5 .
r7 = 2 .
r8 = 4/5 .
x1 = 6 .
x2 = 4 .
```

Fig. 7 The solution to gear design problem.

$$r_5 + r_7 = x_2 \qquad (7)$$
$$r_6 + r_8 = x_2 \qquad (8)$$

From the specification of the sum of the intervals,

$$x_1 + x_2 = 10 \qquad (9)$$

(1), ⋯, (9) are hard constraints. To use standard gears as many as possible, we regard the possible standard values as soft constraints. In other words, we make the following soft constraints to be satisfied as many as possible for every radius r_i:

$$(r_i=1) \lor (r_i=2) \lor (r_i=3) \lor (r_i=4).$$

In this case, the order over solutions is defined by unsatisfied-count-better comparator.

Now, we show how the above problem can be solved by using algebraic CHAL. In algebraic CHAL program, we can use the only constraint symbol, $=$ and the algebraic function symbols such as $+$, $*$, and variables and fractions.

The program in Fig. 6 builds constraint hierarchy of the above problem.

If we ask ?-gear, then algebraic CHAL firstly calculates a set of reduced constraints from hard constraints and then computes all simplified maximal consistent sets of constraints by solving constraint hierarchy. The result after considering soft constraints is shown in Fig. 7. From the result, we cannot make two radii r_6 and r_8 to be standard. This is because hard constraints prevent these gears from being standard gears. Note that if constraints of possible standard values for gear were hard constraints, then we would not get any solution. In CHAL program, thanks to soft constraints, we can get solutions such that radii are standard values as many as possible. In this example, we can make the radii other than r_6 and r_8 to be standard gears.

5. Conclusion

We compare our work with some related researches.

1. Borning et al.[2] were the first to propose the HCLP scheme. However, in Ref. 2), there is no logical formalization of the most preferable solutions. In this paper, we provide a logical formalization of the most preferable solutions for not only locally-predicate-better comparator but also unsatisfied-count-better comparator.
   In Ref. 2), they discuss a relation of HCLP to nonmonotonic reasoning and claim that HCLP can handle the multiple extension problems of nonmonotonic logic. However, our result shows that a constraint hierarchy defined by HCLP is no more than a variant of prioritized circumscription. This means that HCLP can handle only multiple extension problems that can be solved by prioritized circumscription.
2. Baker et al.[1] give a theorem prover of

prioritized circumscription. Since they use the finite domain closure axioms, they impose that their considered domain be finite.
On the other hand, if we use algebraic CHAL, our domain is a complex number. So, semantic restriction does not always impose that the considered domain be finite.

Finally, we summarize the contributions of this paper.

1. We show a logical semantics of constraint hierarchy of HCLP by interpretation ordering in terms of not only locally-predicate-better comparator, but also unsatisfied-count-better comparator.
2. From this semantics, we point out that a solution of constraint hierarchy can be regarded as the most preferable solution defined by semantically-restricted soft constraints. In the semantical restriction, the considered domain is fixed and only a logical combination of domain-dependent constraints can be used.
3. We propose a bottom-up algorithm of computing all maximal consistent constraint sets without any redundant calls of the constraint solver.

Acknowledgments We would like to thank Jun Arima from ICOT, Vladimir Lifschitz from University of Texas at Austin and Yuji Matsumoto from Kyoto University for instructive comments on this paper. Special thanks must go to Hiroyuki Sawada from MEL for tutoring us about a gear-box design.

References

1) Baker, A. B. and Ginsberg, M. L.: A Theorem Prover for Prioritized Circumscription, *Proc. of IJCAI'89*, pp. 463-467 (1989).
2) Borning, A., Maher, M., Martindale, A. and Wilson, M.: Constraint Hierarchies and Logic Programming, *Proc. of ICLP89*, pp. 149-164 (1989).
3) Buchberger, B.: Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory, In Bose, N. ed., *Multidimentional Systems Theory*, pp. 184-232, D. Reidel, Dordecht (1985).
4) Descotte, Y. and Latombe, J.: Making Compromises among Antagonist Constraints in a Planner, *Artif. Intell.*, Vol. 27, pp. 183-217 (1985).
5) Lifschitz, V.: Computing Circumscription, *Proc. of IJCAI85*, pp. 121-127 (1985).
6) Lifschitz, V.: On the Declarative Semantics of Logic Programs with Negation, In Minker, J. ed., *Foundations of Deductive Databases and Logic Programming*, pp. 177-192, Morgan Kaufmann Publishers (1988).
7) Sakai, K. and Aiba, A.: CAL: A Theoretical Background of Constraint Logic Programming and Its Applications, *J. Symbolic Computation*, Vol. 8, pp. 589-603 (1989).
8) Sakai, K. and Sato, Y.: Application of Ideal Theory to Boolean Constraint Solving, *Proc. of PRICAI90*, pp. 490-495.
9) Satoh, K.: Formalizing Nonmonotonic Reasoning by Preference Order, *Proc. of Info Japan 90*, Part II, pp. 155-162 (1990).
10) Satoh, K.: Formalizing Soft Constraints by Interpretation Ordering, *Tran. Inf. Process. Soc. Jpn.*, Vol. 31, pp. 772-782 (in Japanese) (1990), A shortened English version can be found in *Proceedings of the Ninth European Conference on Artificial Intelligence*, pp. 585-590 (1990).
11) Smith, S. F., Fox, M. S. and Ow, P. S.: Constructing and Maintaining Detailed Production Plans, Investigations into the Development of Knowledge-Based Factory Scheduling Systems, *AI Magazine*, Vol. 7, pp. 45-61 (Fall 1986).

Appendix A: An algorithm for solving constraint hierarchy

```
solve_constraint_hierarchy($CH$, $RRC$)
% Solve constraint hierarchy $CH$ with a set of reduced required constraints $RRC$.
begin
  $PA := \{\langle \phi, RRC \rangle\}$
  % $PA$ is a set of pairs of ⟨Combined Constraints, Reduced Constraints⟩.
  for every level $L$ in $CH$ from the strongest to the weakest do
  begin
    if $L \neq \phi$ then
    begin
      for every pair $\langle Cs, RC \rangle$ in $PA$ do
        $NewPA :=$
          maximal_constraints($L, Cs, RC, PA$)
      $PA := NewPA$
    end
  end
  Take every $RC$ of $\langle Cs, RC \rangle$ in $PA$ to form a set, $SC$.
  return SC
end (solve_constraint_hierarchy)
maximal_constraints($L$, $Cs$, $RC$, $PA$)
% Find all maximal subsets in $L$ which is consistent with $RC$.
begin
  $QL := \{\langle Cs, \phi, RC, L\rangle\}$, $NGs := \phi$.
  do
    $QL$, $NGs$, $PA$:=
         maximal_constraints1($QL$, $NGs$, $PA$)
  until $QL = \phi$
  return $PA$
end (maximal_constraints)
maximal_constraints1($QL$, $NGs$, $PA$)
% Produce all extended consistent sets of constraints from $QL$.
% $QL$: a list of quadruple of the following sets of constraints:
% <Combined Constraints, Used Constraints, Reduced Constraints, Rest>
% $NGs$: a set of contradictory combinations of constraints with $RC$.
begin
  $NewQL := \phi$
  for every element $\langle Cs, UC, RC, Rest\rangle$ in $QL$
  do
  begin
    while ($Rest \neq \phi$) do
    begin
      Take one constraint $C$ from $Rest$ and delete $C$ from $Rest$.
      % Note that $Rest$ is decreased by one element for each while loop
      % so that every combination of constraints is checked only once.
      Add $C$ to $Cs$ to get $NewCs$
      % We extend $Cs$ by adding $C$.
      if $C$ is a disjunction then
        for every disjunct $D$ in $C$ do
          $NewQL$, $NGs$, $PA$:=
             maximal_constraints2($D$, $NewCs$,
                 $UC$, $RC$, $Rest$, $NewQL$,
                 $NGs$, $PA$)
      else
        $NewQL$, $NGs$, $PA$:=
           maximal_constraints2($C$, $NewCs$,
               $UC$, $RC$, $Rest$, $NewQL$, $NGs$,
               $PA$)
    end
  end
  return $NewQL$ and $NGs$ and $PA$
end (maximal_constraints1)
maximal_constraints2($C$, $NewCs$, $UC$, $RC$, $Rest$, $QL$, $NGs$, $PA$)
% This is the main procedure of calculating maximal consistent sets of constraints.
begin
  Add $C$ to $UC$ to get $NewUC$.
  if there exists $NG \in NGs$ such that $NG \subseteq$
       $NewUC$ then
    return $QL$ and $NGs$ and $PA$
    % If we see that a subset of $NewUC$ is contradictory then
    % we do not invoke solve and no longer extend $NewUC$.
  $NewRC := \text{solve}(C, RC)$
  % If $C$ and $RC$ is consistent then
  % solve($C$, $RC$) returns a new set of reduced constraints
  % otherwise it returns inconsistent information.
  if $NewRC = inconsistent$ then
  begin
    Add $NewUC$ to $NGs$.
    return $QL$ and $NGs$ and $PA$
    % If we see that $NewRC$ is contradictory then we register it as nogoods
    % and use it for further contradiction detecting and no longer extend $NewUC$.
  end
  if $Rest \neq 0$ then
    Add $\langle NewCs, NewUC, NewRC, Rest\rangle$ to
        $QL$
  if there exists $\langle Cs', RC'\rangle \in PA$ such that
        $NewCs < Cs'$ then
    return $QL$ and $NGs$ and $PA$
    % If we use locally-predicate-better comparator,
    % $NewCs < Cs'$ is $NewCs \subset Cs'$.
    % If we use unsatisfied-count-better comparator,
    % $NewCs < Cs'$ is $|NewCs| < |Cs'|$.
    % If there exists a combined constraints in $PA$
    % which is strictly better than $NewCs$,
    % then $NewCs$ is not a maximal consistent set.
  Delete any $\langle Cs', RC'\rangle \in PA$ s.t. $Cs' < NewCs$.
  % We delete every non-maximal consistent set from $PA$.
  Add $\langle NewCs, NewRC\rangle$ to $PA$.
  return $QL$ and $NGs$ and $PA$
```

end (maximal_constraints2)

Appendix B: Proofs of Theorems

Theorem 1 *M is a most preferable solution w.r.t. $RC(\mathbf{X})$ and the order $<_{\phi}^{lpb}$ if and only if M is a model of the formula (P_{lpb}).*

Proof: We can easily check that $<_{\phi}^{lpb}$ is translated into $(\mathbf{x}\leq\mathbf{X}) \wedge \neg(\mathbf{X}\leq\mathbf{x})$ in (P_{lpb}) by using the translation method defined in Ref. 10). Then, from Corollary 1 in Ref. 10), the set of all the most preferable models w.r.t. $RC(\mathbf{X})$ and $<_{\phi}^{lpb}$ is equivalent to the set of models of the formula (P_{lpb}). □

Theorem 2 *Let A be the axioms of the considered domain and $RC(\mathbf{X})$ be a conjunction of required constraints and $CH(\mathbf{X})$ be a constraint hierarchy. Let $MC_1(\mathbf{X}), MC_2(\mathbf{X}), \cdots, MC_n(\mathbf{X})$ be all maximal consistent sets w.r.t. A and $RC(\mathbf{X})$ and $CH(\mathbf{X})$ for locally-predicate-better comparator. Then, $A \wedge (MC_1(\mathbf{X}) \vee MC_2(\mathbf{X}) \vee \cdots \vee MC_n(\mathbf{X}))$ is logically equivalent to the formula (P_{lpb}). In other words, models of $A \wedge (MC_1(\mathbf{X}) \vee MC_2(\mathbf{X}) \vee \cdots \vee MC_n(\mathbf{X}))$ are exactly all the most preferable models.*

Proof: Let M be a model of $A \wedge (MC_1(\mathbf{X}) \vee MC_2(\mathbf{X}) \vee \cdots \vee MC_n(\mathbf{X}))$. Then, M is a model of A and a model of one of $MC_m(\mathbf{X})$ $(1\leq m\leq n)$. Let $CH^l(\mathbf{X})$ be a set of constraints satisfied in M at a level l $(1\leq l\leq k)$. Then, we can write $MC_m(\mathbf{X})$ as $RC(\mathbf{X}) \wedge CH^1(\mathbf{X}) \wedge \cdots \wedge CH^k(\mathbf{X})$.

Suppose that there is some model M' of $A \wedge RC(\mathbf{X})$ such that $M' <_{\phi}^{lpb} M$ is true. Then, at some level i,

$$\bigwedge_{l=1}^{m_i}((M \models_{\phi} C_l^i(\mathbf{X}))) \bigwedge \neg \bigwedge_{l=1}^{m_i}((M' \models_{\phi} C_l^i(\mathbf{X})) \supset (M \models_{\phi} C_l^i(\mathbf{X})))$$

is true, and for every j $(1\leq j\leq i-1)$,

$$\bigwedge_{l=1}^{m_j}((M \models_{\phi} C_l^j(\mathbf{X})) \equiv (M' \models_{\phi} C_l^j(\mathbf{X})))$$

is true.

Let $CH'^l(\mathbf{X})$ be a set of constraints satisfied in M' at a level l $(1\leq l\leq k)$. The above means that, at the level i, $CH^i(\mathbf{X}) \subset CH'^i(\mathbf{X})$ and for every j $(1\leq j\leq i-1)$, $CH^j(\mathbf{X}) = CH'^j(\mathbf{X})$.

Let $RC(\mathbf{X}) \wedge CH'^1(\mathbf{X}) \wedge \cdots \wedge CH'^k(\mathbf{X})$ be $MC'(\mathbf{X})$. Then, since M' is a model of $A \wedge MC'(\mathbf{X})$, $MC'(\mathbf{X})$ is consistent with A. Therefore, it contradicts the fact that $MC_m(\mathbf{X})$ is a maximally consistent set with A. Thus, M is a most preferable model w.r.t. $A \wedge RC(\mathbf{X})$ and the order $<_{\phi}^{lpb}$. Therefore, a model of $A \wedge (MC_1(\mathbf{X}) \vee MC_2(\mathbf{X}) \vee \cdots \vee MC_n(\mathbf{X}))$ is a model of (P_{lpb}) from Theorem 1.

Let M be a model of (P_{lpb}). Then, M is a most preferable model w.r.t $A \wedge RC(\mathbf{X})$ and the order $<_{\phi}^{lpb}$ from Theorem 1. Let $CH^l(\mathbf{X})$ be a set of constraints satisfied in M at a level l $(1\leq l\leq k)$ and $RC(\mathbf{X}) \wedge CH^1(\mathbf{X}) \wedge \cdots \wedge CH^k(\mathbf{X})$ be $MC(\mathbf{X})$. Suppose $MC(\mathbf{X})$ is not a maximal consistent set. Then, there exists a maximal consistent set of constraints $RC(\mathbf{X}) \wedge CH'^1(\mathbf{X}) \wedge \cdots \wedge CH^k(\mathbf{X})$ where $CH'^l(\mathbf{X})$ is a conjunction of some constraints at a level l $(1\leq l\leq k)$ such that it is consistent with A and satisfies the following condition:

There exists i $(1\leq i\leq k)$ such that
for every j $(1\leq j\leq i-1)$, $CH^j(\mathbf{X}) = CH'^j(\mathbf{X})$ and
$CH^i(\mathbf{X}) \subset CH'^i(\mathbf{X})$.

Let $RC(\mathbf{X}) \wedge CH'^1(\mathbf{X}) \wedge \cdots \wedge CH'^k(\mathbf{X})$ be $MC'(\mathbf{X})$. Since $MC'(\mathbf{X})$ is consistent with A, there exists a model M' of $A \wedge MC'(\mathbf{X})$. Then, M' is a model of $A \wedge RC(\mathbf{X})$ and at the level i,

$$\bigwedge_{l=1}^{m_i}((M \models_{\phi} C_l^i(\mathbf{X})) \supset (M' \models_{\phi} C_l^i(\mathbf{X}))) \bigwedge \neg \bigwedge_{l=1}^{m_i}((M' \models_{\phi} C_l^i(\mathbf{X})) \supset (M \models_{\phi} C_l^i(\mathbf{X})))$$

is true, and for every j $(1\leq j\leq i-1)$,

$$\bigwedge_{l=1}^{m_j}((M \models_{\phi} C_l^j(\mathbf{X})) \equiv (M' \models_{\phi} C_l^j(\mathbf{X})))$$

is true. Therefore, $M' <_{\phi}^{lpb} M$ is true and it contradicts the fact that M is a most preferable model w.r.t $A \wedge RC(\mathbf{X})$ and the order $<_{\phi}^{lpb}$. Thus, $MC(\mathbf{X})$ is a maximal consistent set and M is a model of $A \wedge (MC_1(\mathbf{X}) \vee MC_2(\mathbf{X}) \vee \cdots \vee MC_n(\mathbf{X}))$. □

Theorem 3 *M is a most preferable solution w.r.t. $RC(\mathbf{X})$ and the order $<_{\phi}^{ucb}$ if and only if M is a model of the formula (P_{ucb}).*

Proof: We can easily check that $<_{\phi}^{ucb}$ is translated into:

$$(\langle \mathbf{x}, \mathbf{e}\rangle \leq \langle \mathbf{X}, \mathbf{E}\rangle) \wedge \neg(\langle \mathbf{X}, \mathbf{E}\rangle \leq \langle \mathbf{x}, \mathbf{e}\rangle))$$

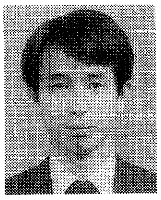
in (P_{ucb}) by using the translation method defined in Ref. 10). Then, from Corollary 1 in Ref. 10), the set of all the most preferable models w.r.t $RC(\mathbf{X})$ and $<_{\phi}^{ucb}$ is equivalent to the set of models of the formula (P_{ucb}). □

Theorem 4 *Let A be the axioms of the considered domain and $B(\mathbf{X}, \mathbf{E})$ be an axiom set for the error function and $RC(\mathbf{X})$ be a conjunction of required constraints and $CH(\mathbf{X})$ be a constraint hierarchy. Let $MC_1(\mathbf{X}), MC_2(\mathbf{X}), \cdots, MC_n(\mathbf{X})$ be all maximal consistent sets w.r.t. A and $B(\mathbf{X}, \mathbf{E})$ and $RC(\mathbf{X})$ and $CH(\mathbf{X})$ for unsatisfied-count-better compar-*

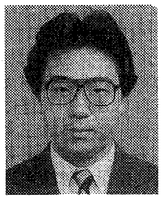
ator. Then, $A \wedge B(\mathbf{X}, \mathbf{E}) \wedge (MC_1(\mathbf{X}) \vee MC_2(\mathbf{X}) \vee \cdots \vee MC_n(\mathbf{X}))$ *is logically equivalent to the formula* (P_{ucb}). *In other words, models of* $A \wedge B(\mathbf{X}, \mathbf{E}) \wedge (MC_1(\mathbf{X}) \vee MC_2(\mathbf{X}) \vee \cdots \vee MC_n(\mathbf{X}))$ *are exactly all the most preferable models.*

Proof: In a similar way to the proof of Theorem 2. □

(Received September 9, 1992)
(Accepted April 14, 1993)



Ken Satoh was born in 1959. He received the B.S. degree in Information science from the University of Tokyo, Japan in 1981. In 1981, he joined Fujitsu Laboratories Ltd. Since 1987, he has been a researcher at the Institute for New Generation Computer Technology (ICOT).



Akira Aiba was born in 1956. He received the Dr. Eng. degree from Keio University in 1986. In 1986, he joined NEC Corporation. Since 1987, he has been working for the Institute for New Generation Computer Technology (ICOT). He conducted research and development of constraint logic programming languages CAL and GDCC at ICOT in the FGCS project. He is currently the deputy manager of the second research department of ICOT. His interests include constraint logic programming and constraint satisfaction. He is a member of the IPSJ.