ICOT Technical Report: TR-0827

TR-0827

# Preference-based Decision Making for Cooperative Knowledge-based Systems

by
Setephen T.C. Wong

January, 1993



ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03)3456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

# Preference-based Decision Making for Cooperative Knowledge-based Systems

Stephen T. C. Wong*
Institute for New Generation Computer Technology (ICOT)

**Abstract**

Recent advances in cooperative knowledge-based systems (CKBS) offer significant promise for intelligent interaction between multiple AI systems for solving larger, more complex problems. In this paper, we propose a logical, qualitative problem-solving scheme for CKBS that uses social choice theory as a formal basis for making joint decisions and promoting conflict resolution. This scheme consists of three steps, namely, (1) the selection of decision criteria and competing alternatives, (2) the formation of preference profiles and collective choices, and (3) the negotiation among agents as conflicts arise in group decision making. In this paper, we focus on the computational mechanisms developed to support steps (2) and (3) of the scheme. In addition, the practicality of the scheme is illustrated with examples taken from a working prototype dealing with collaborative structural design of buildings.

**Categories and Subject Descriptors**: I.2.11 [Artificial Intelligence]: Distributed Artificial Intelligence - coherence and coordination; I.2.1 [Artificial Intelligence]: Problem Solving/Control Methods/and Search - heuristic methods; H.4.2 [Information Systems]: Information Systems Applications - decision support.
**General Terms**: Design, theory, human factors
**Additional Keywords and Phrases**: Social choice theory, decision making, cooperative problem solving, cooperative knowledge-based systems.

*Author's address: S. Wong, ICOT, 21 Fl., Mita-Kokusai Building, Mita, Minato-Ku, Tokyo, Japan 108.

# 1 Introduction

Knowledge-based systems are software systems which apply artificial intelligence (AI) to problem-solving and decision-making tasks. Applications can be found in many areas such as business, science, engineering, and the military. Most knowledge-based systems today are developed to stand alone, thus, their reasoning methods are bounded by a single, monolithic conception of knowledge and action. This inherently limits the scale and the complexity of the problems they can solve.

To solve larger, more complex problems, we must break away from this monolithic conception of problem solving and have knowledge-based systems interact intelligently with each other and/or human users in solving problems beyond their individual scopes. Research into cooperative knowledge-based systems (CKBS) – a group of knowledge-based systems or agents that cooperate with each other and/or human participants to solve problems which require their combined expertise and resources – offer significant promise. Since cooperating agents differ in information content, internal structure, and inference ability, they rarely have consistent viewpoints on mutual problems. Often, agents resort to negotiation with each other in order to reach consensus. Further, the computational speed-up may be a favorable side-effect, but is not the primary objective of cooperative knowledge-based systems. This sets apart projects of cooperative computing from other projects that solely aim to increase the computation speed through parallel or distributed processing of AI programs.

Experimentation dominates current CKBS research. This indicates that this research is in its infancy. Recent surveys on the activities of this field are in [16, 12]. In this paper, we propose a new cooperative problem solving scheme that uses the social choice theory as a *formal* basis for making joint decisions and promoting conflict resolution. In particular, we focus on the logical, qualitative aspects of cooperative decision making.

This work is application-driven. The scheme originated from efforts in the NSF Center of Advanced Technology for Large Structural Systems (ATLSS) at Lehigh University to study the principles and theories behind cooperative problem solving and to use them to guide the systematic development of computer-integrated construction systems [2]. The scheme has been implemented in a distributed knowledge-based system prototype, Building Design Network (BDN), for the preliminary structural design of buildings by three human participants: the designer, the fabricator, and the erector, who make joint design decisions [29]. Roughly, a preliminary design problem can be described as the tasks of generating several design alternatives and of selecting the best of these competing alternatives. Preliminary design exhibits many unfavorable conditions for applying prevalent decision theories.

First, credible, well-behaved quantitative data about competing alternatives is difficult to obtain in practice, and such data is unlikely to be applicable after the addition of new facts or rules into the existing knowledge bases. Most decision theories, such as utilities theory [33], fuzzy logic [34], evidence theory [26], and probabilistic reasoning [22], however, depend on the availability of such data to various degrees. Second, a design decision is ideally a collective one involving participants with different responsibilities and specialities. In contrast, most decision theories focus on decisions involving the choices of only one agent or the outcomes determined by that agent's choices and background knowledge. Third, agents and participants are required to work together to make collective choices. This, incidentally, differs from competitive game theory [8, 33]. Games are decision-making situations that always involve more than one agent, but they do not count as group decisions because each agent chooses an action with the aim of furthering individual goals, not mutual goals. A further distinction is that the cooperating agents are sincere in sharing knowledge, whilst the competing agents tend to hide their reasoning and motives from one another.

BDN agents are physically connected by a network and communicate with each other by passing messages. Each of these agents contains knowledge on a speciality, that is, design, fabrication, or erection of the building. Each of them also interacts frequently with its participant as well as other agents to solve the problems together, either in real time or by electronic mail. The knowledge base of every agent is written predominately in Prolog [4] and is partitioned into four knowledge modules that can be run concurrently. The functions of these modules are briefly described as follows:

- The base-level knowledge module models two things: (i) the local agent's expertise, preferences, and inference rules, and (ii) the physical components and properties of the building structure to be designed. The knowledge in this module is represented using an object-based semantic network [30, 29], which encompasses production rules, relations, and objects in its data structures.

- The strategic knowledge module includes a set of problem solving strategies and methods at both local and group levels. The preference scheme discussed in this paper resides in this module. This module and the base-level knowledge module form the core reasoning engine of the local agent and operate as one computational process.

- The communication knowledge module functions as a knowledge server between the local agent and other agents in the network. It contains communication strategies that keep track of message transactions with remote agents and govern the interaction between agents, e.g.,

to whom to send a message and which message to interpret first when multiple messages are received. It also has translation rules that convert the content of external messages into an internal data structure, and vice versa. This module is an independent computational process. The details of this module are described in [31].

- The user-interface knowledge module contains facilities that support a variety of ways for the user to inspect the knowledge represented and to observe the information processed. These include: the display of the inference steps of the agent, the iconic and graphical representations of the knowledge base for direct manipulation, and the backtracking of the justifications of the derived conclusions. This module of the local agent runs separately from the other three modules of the agent. The paradigm for constructing the interface facilities can be found in [32, 29]

The organization of the remainder of this paper is as follows. Section 2 outlines the operational steps of the decision-making scheme. Sections 3 and 4 present the constraints and strategies developed to derive preference orderings of design alternatives. In Section 5, we describe the supporting facilities for the negotiation of preferences between agents. The last section concludes this work and mentions possible future work.

## 2 Cooperative Decision Making Steps

This scheme considers the problem of cooperation, such as in building design, as a decision-making problem. As shown in Figure 1, the scheme tackles such a problem with three operational steps, namely, identification, processing, and negotiation. In this figure, the arrows indicate the flow of information, and the symbol $\oplus$ means that two or more knowledge modules together derive one or more outcomes directly. We describe each of these steps in the following subsections:

### 2.1 Identification

This step consists of (i) identifying a decision agenda which contains a set of criteria, the order of their importance, and the names of agents to which criteria are of concern; (ii) selecting a set of competing alternatives; and (iii) deriving a set of ordering relations between some pairs of these alternatives. Criteria are defined as the particular aspects of the alternatives that are correlated with the desired outcome of the alternatives. They are different from objectives. The latter are directly connected with the desired outcomes, but the estimation of these will be generally fuzzy. Consider, as an example, the design of beam-column connections for a building. Let the desired

outcome be a very profitable project. Correlated with this outcome, "having a fast return on investment" may be the first financial objective. The statement, "the designer knows that welding for building connection is cheaper than bolting," is a criterion concerning the cost between two competing methods of connection: welding and bolting. But because its evaluation will be more accurate than the fuzzy evaluation of the financial objective (fast investment return), it is likely to be more important in decision making.

Moreover, criteria are of concern to individual agents of particular specialities, i.e., they are evaluated according to individual perspectives. Considering the connection design of buildings, a designer would be interested in criteria such as strength and stiffness of connections; a fabricator would be concerned with the difficulties and costs of fabricating parts; and the erector would worry about the labor costs and safety when erecting the structure in the field.

Agents derive competing alternatives of a problem from individual base-level knowledge modules. Depending on the adequacy of these knowledge modules, sometimes, the cooperating agents may have to communicate with each other to identify a common set of competing alternatives. In contrast to the use of cardinal ratings in most decision theories, an agent applies the heuristic rules of its knowledge base to derive some initial preference expressions for its criteria. These expressions are ordering relations for pairs of alternatives. We consider two types of ordering relations, that is, for two alternatives $x$ and $y$, an agent either prefers $x$ to $y$, $y$ to $x$, or is indifferent between $x$ and $y$. Moreover, we assume that preference and indifference are mutually exclusive and exhaustive for any set of competing alternatives. These expressions provide a compact way to represent individual viewpoints on various criteria.

If one takes only the viewpoint of social choice theory, cardinal ratings would work better than preference expressions. Furthermore, these binary, ordinal expressions can be derived from cardinal ratings. In the context of cooperative decision making, however, the real matter is not whether ordinal relations or cardinal ratings would work better. It is whether the expressions or ratings can be systematically derived, and at the same time, their justifications can be readily retrieved during cooperation. The reason is that, for non-trivial problems, it is unlikely that the knowledge of cooperating agents is complete and is consistent with one another. During the operation, agents would have to negotiate over possible conflicts and to obtain new or missing information from the human participants. In other words, one should avoid the black box approach to cooperative decision making.

To further illustrate the point of explicit background knowledge, we sketch, in the following, certain shortcomings of an earlier attempt by our center to tackle a simpler design problem of BDN

using pre-defined cardinal ratings.

- Extensibility

  The ratings of the earlier prototype are *magic* numbers provided by the domain experts. This works reasonably well for a few alternatives, but does not scale up. The domain experts are unable to come up with credible ratings for a larger set of alternatives.

- Explanation

  The ratings are *pre-defined*. These ratings give the potential users a false sense of security, because it is impossible to capture the design intention with a set of fixed numbers on the alternatives. The real-world users would demand the justifications of any choice beyond ratings.

- Flexibility

  Design is an open problem, and the priorities of design alternatives would vary with situations. The pre-defined ratings lack the flexibility to deal with realistic design problems. In contrast, the proposed scheme provides a structure for the users to retrieve the background justification of an expression and to discuss the credibility of that justification for updating purposes. The updating of background knowledge is outside the scope of this paper, however, Section 4 discusses the revision of preferences.

Our experience indicates that the knowledge needed to derive binary, ordering relations is much easier to acquire than the knowledge to assign credible, quantitative data. This motivates the use of preference expressions in our scheme.

## 2.2 Processing

Figure 2 illustrates a more detailed information flow of the processing step. A processing step consists of two parts. In the first part, for every criterion, a complete set of preference expressions is deduced. that is, all pairs of competing alternatives are related. This complete set enables the formation of a linear ordering of alternatives regarding that criterion. The formulation of an individual ordering is based on initial preference expressions (generated in the identification step) and certain properties or constraints of these expressions such as symmetry, transitivity, and reflexivity. As an example, suppose that $x, y, z$ are members of competing alternatives of a criterion. If an agent prefers $x$ to $y$ and $y$ to $z$, then it prefers $x$ to $z$ by transitivity. Or, if another agent is indifferent between $x$ and $y$ but prefers $y$ over $z$, then one can say that it should also prefer

$x$ over $z$. The set of individual orderings of all criteria – a preference profile – reflects distinct, and sometimes conflicting, viewpoints of the agents over the outcome.

Thus, preference constraints provide a logical way to derive missing information from an initial set of preferences. This differs from quantitative methods such as cardinal ratings in which unknown ratings are often not derivable from known ones. Sometimes, the preference expressions of certain criterion may remain incomplete even through all known preference consequences are deduced. This scheme resorts to certain heuristic to complete the set.

In the second part of the processing step, the set of individual orderings is combined to form an aggregated ordering. The top ranked alternative(s) of the ordering thus constitute a recommended solution. Such a solution is called a collective choice in the social choice theory [1, 24]. Aggregation is normally involved in the social choice literature with the objective of finding an optimal choice. Many sophisticated mathematical techniques, particularly in operation research, have been developed to attain this goal.

The preference scheme, however, *embarks* on a different path from this tradition. Instead, it proposes to use the result of aggregation mainly as feedback to the cooperating agents, who then decide whether to endorse the choices or to negotiate over conflicts. For applications which do not have established methods of aggregation (which is likely to happen as little is known about the cooperative behavior of several agents with different specialities and perspectives), the use of a default method is necessary. There are many standard aggregation methods [25]. One of them, the simple majority rule which states that all criteria are of equal weight and thus have equal votes, is adopted as a default method in the scheme. This method, however, does not imply that all agents have equal voting power. The voting power of an agent on a particular problem is proportional to the number of its criteria listed in the problem agenda. In addition, when the criteria are of different importance, the weight majority rule is applied instead [25].

The preference profiles and aggregated orderings are common knowledge among all agents. Every agent allocates a working space on its terminal screen to display and manipulate such common knowledge in real time. This interface is essential in the negotiation step, which is discussed next.

### 2.3 Negotiation

By negotiation, we mean a discussion in which the intended agents exchange information and come to an agreement over conflicts. The negotiation step is essential for cooperative problem solving from both the standpoints of social choice theory and knowledge-based systems.

First of all, in social choice literature, every existing aggregation method is bound to face some technical problems and controversy. A negotiation step is included to allow the agents to discuss with each other as such disputes arise. Second, the explicit representation of all relevant knowledge into computational agents is more a working hypothesis than a reality. As a whole, the agents would only contain partial knowledge of the problem domain. Further, the knowledge of these agents is normally acquired from experts of different specialities. Hence, it is likely that the agents would have conflicting opinions and knowledge. In this respect, the purpose of negotiation on preferences is to compensate for the incompleteness and inconsistency of knowledge by holding discussions with a focused agenda.

In cooperative problem solving, it is assumed that the conflict is good. It is assumed that agents, when faced with conflict, will share information about their preferences, about why they have these preferences, and then search out other alternatives which satisfy the criteria of as many agents as possible. The result will be a high quality decision from the point of view of the overall problem.

In BDN, an agent initiates a negotiation session whenever its index of negotiation exceeds the threshold value. (Scheduling is required if more than one agent complain at the same time. Needless to say, the users can also initiate negotiation sessions.) The process of negotiation, however, is driven by human participants and thus is not automatic. Our design philosophy is to assist the users to make better joint decisions, not to dictate the solutions of decision problems.

Suppose that a participant disagrees with a collective choice. He can probe into the preference profile to find disagreeable preference expressions, query particular agents for justifications, discuss with other participants over the network, and reach an agreement or compromise using certain resolution strategies. As a result, the participants may also modify the knowledge bases of their agents in the process. (Again, only the revision of preference expressions and orderings is discussed in this paper.) A new aggregated ordering and a new collective choice are derived after every negotiation session. Any agent still not satisfied with the choice (measured by its new negotiation index) can initiate another session of negotiation. Since the breaking off of negotiation is considered non-cooperative, we shall not be concerned with this behavior here.

Effective strategies for conflict resolution are needed in negotiation. Usually, these strategies depend on the applications and organizational structures [20, 17, 18]. In BDN, the predominant mode of conflict resolution is the use of bargaining or compromise, i.e., the participants push for acceptance of the alternative which is preferred by their criteria and occasionally *give in* by making incremental changes to their preferred alternatives. Occasionally, forcing is used to back up the

bargaining approach when lack of agreement stymies the group, i.e., a participant may have a position of power or knowledge to force a preferred alternative on the rest of the group.

Generally speaking, the forcing of agreement is one of the cardinal sins of social choice theory. However, when a decision must be reached after a long deadlock in negotiation, it is a common practice in human organizations for the one who is accountable to make the final choice. In the domain of building design, the designer is the one who is accountable for the entire building structure by law. Thus, the designer is given the authority to force agreement in BDN.

## 3 Preference Constraints

Let $S$ be a set of alternatives for a problem. We use the expression $P(a,i,x,y)$ to mean "agent $a$ prefers $x$ to $y$ in criterion $i$" and $I(a,i,x,y)$ to mean "agent $a$ is indifferent between $x$ and $y$ in $i$." The preference and indifference constraints of the scheme are stated explicitly as follows:
For any $x,y,z \in S$, any agent $a$, and any criterion $i$,

**Constraint 1** *(Reflexive and Symmetry)*

1. $I(a,i,x,x)$
2. $I(a,i,x,y) \supset I(a,i,y,x)$

**Constraint 2** *(Asymmetry)*

1. $P(a,i,x,y) \supset \neg P(a,i,y,x)$
2. $P(a,i,x,y) \supset \neg I(a,i,x,y)$
3. $I(a,i,x,y) \supset \neg P(a,i,x,y)$
4. $I(a,i,x,y) \supset \neg P(a,i,y,x))$

**Constraint 3** *(Connectivity)*

1. $P(a,i,x,y) \vee P(a,i,y,x) \vee I(a,i,x,y)$

**Constraint 4** *(Transitivity)*

1. $I(a,i,x,y) \wedge I(a,i,y,z) \supset I(a,i,x,z)$
2. $P(a,i,x,y) \wedge P(a,i,y,z) \supset P(a,i,x,z)$
3. $P(a,i,x,y) \wedge I(a,i,y,z) \supset P(a,i,x,z)$
4. $I(a,i,x,y) \wedge P(a,i,y,z) \supset P(a,i,x,z)$

The scheme is implemented in BDN using the logic programming language Prolog. For generality, we describe the preference scheme in formal logic. The heuristic rules for deriving preferences

and the set of preference constraints are indeed analogous to the Horn clauses of logic. Note that we say 'analogous', not 'equivalent', because of the order in which predicates are evaluated and because of possible side-effects of Prolog computation.

Basically, the constraints say that (i) preference is irreflexive, asymmetric, and transitive; (ii) indifference is reflexive, symmetric, and transitive; and (iii) preference (for $x$ over $y$ or for $y$ over $x$) and indifference are mutually exclusive and exhaustive. These constraints are not independent as asymmetry and connective constraints are interrelated. For practical purposes, however, they are made explicit in the discussion.

The connectivity constraint states that any two competing alternatives $x$ and $y$ must be related in one preference expression but not both. We regard such a constraint as an ideal which we should aim for in making important decisions. Nevertheless, humans fail from time to time to have connected preferences; especially when many alternatives must be evaluated at once and our concentration becomes overtaxed. Such a phenomenon can be explained by the structure of human memory [13]. It has long been known that the human brain possesses a short-term memory that allows the storage, for a limited time, of a limited number of items for immediate use. We also possess a long-term memory, which stores far more items in a lifetime. However, when these data have to be retrieved, it is through a linear chain of associations, and the process for retrieval can be relatively long and somewhat painful. Try, for instance, to remember a birthday once forgotten.

Often, the quantity of information required in making complex decisions is too large to allow simultaneous treatment by short-term memory. There is no clearly dominant method allowing us to retrieve information quickly. That is why people tend to confine their decision choices to a small set of alternatives. Maybe our inability to solve bigger problems is largely due to the limitations of short-term memory. Representing large amounts of rules and facts in knowledge-based systems with a fast retrieval information capability is a promising means to alleviate such decision problems.

Transitivity holds for all competing alternatives. For example, if one prefers ice tea to a milk shake and a milk shake to ice cream, then one prefers ice tea to ice cream. Or in choosing a graduate school, if one prefers computer science to law and is indifferent between law and medicine, then one prefers computer science to medicine. Humans, however, do not always have transitive preferences. When one adds small amounts of sugar to successive cups of coffee in such a way that the increments in sugar levels between adjacent cups are not detectable, then one can set up a sequence of cups of coffee with this property: people will be indifferent between adjacent cups, but not between the first and last. Does this mean that the assumed constraint of the transitivity of indifference is false?

The point, however, is that the transitivity constraints characterize the preferences of an ideally rational agent. Humans fail the constraints because we fall short of that ideal, by lacking sufficiently refined tastes to detect the difference of "sweetness" between adjacent cups as in the above example. But rather than dismiss the transitivity constraints, we should take steps, when a decision is important, to emulate ideal agents more closely. For example, we could use chemical tests to determine the relative sweetness, if the decision is important enough to be treated carefully. Thus, we should try for transitive preference orderings when and where we can in designing intelligent systems, because transitive preference orderings organize preferences into a simple and tractable structure.

A preference ordering is the ranking of equivalent classes of alternatives with respect to $I$. These equivalent classes are also called indifference classes. They are so called because the agent is indifferent to alternatives in the same classes but prefers alternatives in one class to those in a different class. That is, one indifference class ranks above another if its members are preferred to those of the other. The following example illustrates this.

**Example 3-1**

Suppose agent $a$'s preferences among six alternatives – $x_1, x_2, x_3, x_4, x_5$, and $x_6$ – with respect to criterion $i$ are: $I(a,i,x_1,x_2), P(a,i,x_2,x_3), P(a,i,x_3,x_4), I(a,i,x_4,x_5)$, and $I(a,i,x_5,x_6)$.
The transitivity constraints permits the agent to derive additional information from the preferences given. For example, Constraint 4.1 yields $I(a,i,x_4,x_6)$ and Constraint 4.4 yields $P(a,i,x_1,x_3)$. This additional information tells the agent that the alternatives divide into the following ranked (linear ordered) indifference classes: (1) $<x_1,x_2>$, (2) $x_3$, (3) $<x_4,x_5,x_6>$. The notation $<x,y>$ means that $x$ and $y$ are of the same equivalent class with respect to $I$. Note that only positive preference expressions are used in deriving the preference ordering.

# 4 Preference Sets

This section presents the static and dynamic properties of the preference sets and discusses an aggregation method of preferences.

## 4.1 Static Properties

Let $M$ be a set of preference expressions for pairs of alternatives in $S$ of agent $a$ and criterion $i$. Let $K(a,i,x,y)$ stand for either $P(a,i,x,y)$ or $I(a,i,x,y)$. The set of all deductive consequences of $M$,

$K(a,i,x,y) : M \vdash K(a,i,x,y)$, is denoted as $Cn(M)$ and is called the preference set of $M$. Thus, $Cn(M)$ contains all the preference expressions that the agent knows. (One motivation for adopting Prolog as the implementation language is its built-in deductive inference.)

A preference set has the following closure properties: (i) $M \vdash \alpha$ if and only if (iff) $\alpha \in Cn(M)$, and (ii) $Cn(Cn(M)) = Cn(M)$. An agent $a$ accepts an expression $K(a,i,x,y)$ iff $K(a,i,x,y) \in Cn(M)$; rejects $K(a,i,x,y)$ iff $\neg K(a,i,x,y) \in Cn(M)$. Otherwise, $K(a,i,x,y)$ is an unknown expression for $a$. In addition, $Cn(M)$ is inconsistent iff $K(a,i,x,y)$ is both accepted and rejected by $a$, that is, when both $K(a,i,x,y)$ and $\neg K(a,i,x,y)$ are members of $Cn(M)$. It is important for an agent to maintain the consistency of its preference set at any moment, otherwise, whatever conclusion it derives from the set would be meaningless. However, it is inefficient to detect inconsistencies by searching for the negation of any of its preference expressions during the computation. The following property offers an alternative, but effective, means for detecting preference inconsistency during inferential process.

**Property 1** *(Consistency)*
*$Cn(M)$ is inconsistent at time $t$ iff $P(a,i,x,x) \in Cn(M)$ at $t$.*

Proof:
(if): First, consider the case where $P(a,i,x,y)$ is both accepted and rejected in $Cn(M)$, then, $P(a,i,x,y)$ and $\neg P(a,i,x,y)$ are in $Cn(M)$. By asymmetry constraints, either $P(a,i,y,x)$ or $I(a,i,y,x)$ is present in $Cn(M)$. By transitivity constraints, $P(a,i,x,y) \wedge P(a,i,y,x) \supset P(a,i,x,x)$ and $P(a,i,x,y) \wedge I(a,i,y,x) \supset P(a,i,x,x)$. Consider the next case that $I(a,i,x,y)$ is both accepted and rejected in $Cn(M)$, then $I(a,i,x,y)$ and $\neg I(a,i,x,y)$ are in $Cn(M)$. $\neg I(a,i,x,y)$ must be derived from $P(a,i,x,y)$ by the asymmetry constraints. But we know $I(a,i,x,y) \supset I(a,i,y,x)$ and by transitivity, $P(a,i,x,y) \wedge I(a,i,y,x) \supset P(a,i,x,x)$.
(only if): By asymmetry, $P(a,i,x,x) \supset \neg P(a,i,x,x)$ such that $\neg P(a,i,x,x) \in Cn(M)$. Hence $Cn(M)$ is inconsistent.

Making use of Property 1, an agent can easily spot inconsistencies in its preference set during the computation once it matches the pattern $P(V,V)$, where $V$ is a variable, with a preference expression in its preference set.

### 4.2 Dynamic Properties

So far we have treated the statics of the preference set and we now turn to the dynamics: updating preference sets. In this subsection, we present some key properties used in guiding the updating

of preference sets and orderings. For clarity, we will skip trivial proofs. There are three kinds of change to a preference set: (i) expansion – a new preference expression (and its consequences) is added to the preference set without retracting any of the old preferences; (ii) contraction – a preference expression is retracted from the preference set without generating any new preferences; and (iii) revision – a positive preference expression between two alternatives x and y is changed to another positive expression between x and y.

A common property of expansions and contractions is what Gardenfors calls consistent changes of a state of a belief [15]. In this scheme, a belief refers to a preference expression. That is, there is no preference expression that is accepted in an expansion or that is retracted in a contraction which contradicts any expressions in the earlier preference set. The revision, however, is not a consistent change.

Some dynamic properties of preference sets are stated as follows. For any competing alternatives $x_1, x_2, y_1, y_2 \in S$ and an initial set of preference expressions $M$ of agent $a$ concerning an criterion $i$, let us denote $A$ and $B$ as any two preference expressions $K_1(a, i, x_1, y_1)$ and $K_2(a, i, x_2, y_2)$. We define $Cn_A^+(M)$ as the logical consequence of $M$ together with $A$, i.e., $Cn_A^+(M) = Cn(M \cup \{A\}) = \{B : M \cup A \vdash B\}$. An expansion adds a new expression into the preference set but does not remove any other expression. Thus, this kind of change is monotonic.

**Property 2** *(Expansion)*

1) *For any* $M, Cn(M) \subseteq Cn_A^+(M)$.

2) *For any consistent preference set* $Cn(M), Cn_A^+(M)$ *is consistent iff* $\neg A \notin Cn(M)$.

3) *For any consistent preference set* $Cn(M)$, *if both* $A$ *and* $B$ *are undecided in* $Cn(M)$, *then* $Cn_{A\&B}(M) = Cn(M \cup \{A\} \cup \{B\})$ *is possibly inconsistent.*

   *Proof:*

   *A simple illustration helps to prove the lemma. Consider three alternatives* $x, y,$ *and,* $z$ *in Figure 3.a where* $Cn(M)$ *consists of* $P(a, i, x, y)$ *only, as indicated by an arrow from* $x$ *to* $y$. *Both* $P(a, i, y, z)$ *and* $P(a, i, z, x)$ *are not known in* $Cn(M)$. *Adding* $P(a, i, y, z)$ *and* $P(a, i, z, x)$ *simultaneously to* $Cn(M)$ *renders the expanded preference set inconsistent, that is,* $P(a, i, x, y) \wedge P(a, i, y, z) \supset P(a, i, x, z)$ *and* $P(a, i, x, z) \wedge P(a, i, z, x) \supset P(a, i, x, x)$. *In fact, one can see the circularity of ordering relations among* $x, y,$*and* $z$ *in Figure 3.b.*

4) *For any consistent preference set* $Cn(M)$, *if* $\neg A \notin Cn(M)$, *then* $Cn_A^+(M) = Cn_A^+(Cn(M))$.

   *Proof:*

   $Cn_A^+(Cn(M)) = Cn(Cn(M) \cup \{A\})$ *must be a set larger or equal to* $Cn(M \cup \{A\})$ *since the*

*initial set of preference expressions of the former includes that of the latter, i.e.,* $M \cup \{A\} \subseteq Cn(M) \cup \{A\}$. *To prove the theorem it is sufficient to show that whatever is in* $Cn_A^+(Cn(M))$ *must also be in* $Cn_A^+(M)$.

*Consider* $B \in Cn_A^+(Cn(M))$. *Suppose* $B \in Cn(M) \cup \{A\}$, *then either* $B \in Cn(M)$ *or* $B = A$. *In both cases,* $B \in Cn(M \cup \{A\})$. *Then, let us assume* $B \notin \{Cn(M) \cup \{A\}\}$ *such that there exists* $C \in Cn(M)$ *such that* $C$ *and* $A$ *jointly entail* $B$ *by transitivity. Since* $C \in Cn(M \cup \{A\})$ *and* $A \in Cn(M \cup \{A\})$, *we, again, have* $B \in Cn(M \cup \{A\})$. *Contradiction.*

Denote $Cn_A^-(M)$ as the contraction of a preference set $Cn(M)$, by whatever process is used, with respect to $A$, then:

**Property 3** *(Contraction)*

*1)* $Cn_A^-(M) \subseteq Cn(M)$.

*2) If* $Cn(M)$ *is consistent, then* $Cn_A^-(M)$ *is consistent.*

The main problem concerning contractions of preference sets is that, when retracting an expression $A$ from a preference set $Cn(M)$, there may be other expressions in $Cn(M)$ that either separately or jointly entail $A$. If an agent wants to keep the contracted preference set closed under deductive consequences, it is necessary to give up other preference expressions as well. For example, if $P(a,i,x,z)$ is accepted in $Cn(M)$ just because it is a deductive consequence of $P(a,i,x,y)$ and $P(a,i,y,z)$, which are also in $Cn(M)$, then either $P(a,i,x,y)$ or $P(a,i,y,z)$ must be rejected in retracting $P(a,i,x,z)$. The problem is then to determine which one should be given up and which should be retained.

Suppose that agent $a$ has a consistent preference set $Cn(M)$ with respect to criterion $i$. It wants to retract $K(a,i,x,y)$ from the set. The strategy of contraction used to derive $Cn_{K(a,i,x,y)}^-(M)$ is a heuristic backward search based on the conservation principle of knowledge.

**Strategy 1** *(Contraction algorithm)*

*1) Retract* $K(a,i,x,y)$.

*2) If there exists* $A$ *and* $B$ *in the current preference set such that* $A \wedge B \supset K(a,i,x,y)$, *then go to step 3; otherwise, exit*

*3) Denote the current initial set of preference expressions as $M$, if $A, B \in M$, i.e., they are initial expressions, then step 3.a, else step 3.b*

*(a) either ask the user which one to retract or do the following*

*if $A$ is a $P$ expression and $B$ is an $I$ expression, then retract $B$ and its symmetrical counterpart from $M$*

*else if $A$ and $B$ are both $I$ expressions, then randomly pick one to retract together with its symmetrical counterpart*

*else if $A$ and $B$ are both $P$ expressions, then randomly pick one to retract;*

*denote the contracted initial preference set as $M'$;*

*(b) if $A \notin M$ and $B \in M$, then mark $B$*

*else if $A \notin M$ and $B \notin M$, then randomly mark one;*

*repeat step 2 for retracting the unmarked preference expression.*

*4) Delete $Cn(M)$, unmark any previous expressions, and generate the set $Cn(M')$ from $M'$.*

*5) Repeat step 2.*

In retracting $K(a, i, x, y)$, agent $a$ searches backward to find any initial expressions that entail $K(a, i, x, y)$. If there is only one such expression, then the agent retracts it. If there are two, then the agent has to decide which one to retract. To do so, it either queries the user for advice or adopts a default strategy that puts $P$ at a higher priority than $I$ (see step 3.a of Strategy 1). The justification is that $P$ has a better discriminatory power in deriving preference orderings. Another rationale is that things are not really indifferent, just that there is insufficient information to distinguish them yet. The following example shows how the strategy operates.

Example 4-1

Consider Example 3-1 again, where

$M = \{I(a, i, x_1, x_2), P(a, i, x_2, x_3), P(a, i, x_3, x_4), I(a, i, x_4, x_5), I(a, i, x_5, x_6)\}$

and the indifference classes are ranked in the following order: $< x_1, x_2 >, x_3, < x_4, x_5, x_6 >$. Suppose now that agent $a$ wants to retract the preference expression $P(a, i, x_3, x_6)$ from its preference set. When applying Strategy 1, the nested steps of the operation are as follows:

Level 1

Step 1: Retract $P(a, i, x_3, x_6)$.

Step 2: Find $P(a,i,x_3,x_4) \wedge I(a,i,x_4,x_6) \supset P(a,i,x_3,x_6)$

Step 3.b: $P(a,i,x_3,x_4) \in M, I(a,i,x_4,x_6) \notin M$, mark $P(a,i,x_3,x_4)$ and attempt to retract $I(a,i,x_4,x_6)$

Level 2

Step 2: Find $I(a,i,x_4,x_5) \wedge I(a,i,x_5,x_6) \supset I(a,i,x_4,x_6)$

Step 3.a: $I(a,i,x_4,x_5) \in M, I(a,i,x_5,x_6) \in M$. Suppose the user retracts $I(a,i,x_5,x_6)$. Then, $M' = \{M - \{I(a,i,x_5,x_6)\} - \{I(a,i,x_6,x_5)\}\}$

Step 4: Generate $Cn(M')$.

Step 5: Repeat step 2.

Level 3

Step 2: There exists no $A \wedge B \supset P(a,i,x_3,x_6)$. Exit.

Thus, $Cn(M')$ is the contracted set.

To retract a negated preference expression such as $\neg P(a,i,x,y)$, one should notice that the only preference constraint that deduces the negative expression is the asymmetry constraint $P(a,i,y,x) \supset \neg P(a,i,x,y)$. One thus uses Strategy 1 to retract $P(a,i,y,x)$. The contracted set will exclude $\neg P(a,i,x,y)$ as desired. Also, the agent will compute a new contracted preference set every time it retracts an initial preference expression. For computational efficiency, one may refrain from computing the contracted set until all preference expressions that entail $K(a,i,x,y)$ have been retracted. In this way, the agent only needs to compute preference consequences once. Knowledge, however, is generally not easily acquired. Unnecessary loss of knowledge is therefore to be avoided. The agent should try to retain as many old preference expressions as possible when retracting. This argument is better illustrated in the following example.

### Example 4-2

It is more intuitive to describe the precedence relations of preferences graphically. Consider the graph shown in Figure 4.a, where each node represents a preference expression, arrows indicate the precedence relations of deduction, and an arc imposes a joint entailment. For example, $r_2$ and $r_3$ jointly entail $r_5$, that is, $r_2 \wedge r_3 \supset r_5$, while $r_6$ directly deduces $r_8$, that is, $r_6 \supset r_8$. In the figure, the initial set of preference expressions $M = \{r_1, r_2, r_3, r_4\}$. We compare two cases of retracting $r_8$ here, the first one follow is Strategy 1 and the second uses the suggested method.

(i): Retract $r_8$. Then, suppose the agent takes a backward path $r_6 \supset r_8$ such that it reaches $r_2$ and

$r_3$ and decides to retract $r_3$. The contracted set derived excludes $r_8$ (one can see from Figure 4.a that $r_1, r_2$, and $r_4$ alone do not entail $r_8$), and, thus, ends the contraction process.
(ii): Retract $r_8$. Following the same backward path $r_6 \supset r_8$ as in (i), the agent first retracts $r_3$. Then, it takes another backward path $r_7 \supset r_8$, and subsequently $r_4 \wedge r_5 \supset r_7$. Since $r_3$ has been retracted in the previous round, $r_5$ is assumed to be an initial expression and the agent has to retract either $r_4$ or $r_5$. Say, the agent retracts $r_4$. The subsequent backward searches will lead the agent to retract $r_1, r_6$ and $r_7$. The numbers in Figure 4.b indicate the sequence of retraction. Surely $r_8$ is not in the contracted set, but only $r_2$ is left in the set!

We consider that the conservation of knowledge is more important than the gain in computational speed. One may worry, however, that the penalty on the computation speed, when deducing a preference set of a large number of alternatives, would yield unacceptable performance for practical use. The implemented prototype took only a few seconds to generate a preference set involving about a hundred alternatives in SUN SPARC workstations. Some programming tricks can also be applied to avoid direct application of certain constraints. For example, BDN agents only generate partial preference sets but can deduce missing preference expressions dynamically in one or two inference steps. This cuts the computation time by more than half compared to the time required to generate a full set. A description of such implementation details, however, would be inappropriate in this forum.

In addition, we doubt that any practical problem would have anywhere near a hundred competing alternatives. Problems of such a scale are normally first decomposed into simple, manageable subproblems and then solved individually or iteratively. This is the case for BDN application.

Revision is nonmonotonic - a new preference is added, but not all the old ones are retained. In the negotiation, distributed agents are often called upon to revise their individual preferences. One of the main reasons for this is that many of the things the agents accept as certain may not be well founded - they may believe things because of prejudice, because of a lack of relevant information, or because of faulty inference. Communicating intents and sharing information between agents helps to alleviate many of these deficiencies.

This scheme considers a revision as a two-step operation: retract a previous expression of the preference set and add a new one to the contracted set. Denote $Cn^*_{A,B}(M)$ as the revision preference set obtained by retracting expression $B$ but adding expression $A$, then we have: $Cn^*_{A,B}(M) = Cn^+_A(Cn^-_B(M)) = \{C : Cn^-_B(M) \cup A \vdash C\}$.

Property 4 *(Revision)*

*1) Let* $Cn(M)$ *be consistent.* $Cn^{*}_{A,B}(M)$ *is consistent iff* $\neg A \notin Cn(M)$.

*2)* $Cn^{-}_{A}(Cn^{+}_{A}(M)) = Cn(M)$ *if* $A$ *is undecided in* $Cn(M)$.

*3)* $Cn^{+}_{A}(Cn^{-}_{A}(M)) \neq Cn(M)$, *except when* $A$ *is the only expression retracted in* $Cn^{-}_{A}(M)$.

*Proof:*

*$A$ is unknown or $A$ is false in $Cn(M)$ gives $Cn^{-}_{A}(M) = Cn(M)$ or $Cn(M) \subseteq Cn^{+}_{A}(Cn^{-}_{A}(M))$. Consider that $A$ is true in $Cn(M)$ and there are two possible cases. First, suppose that $A$ is true in $Cn(M)$ and there exists some other preference expressions entail $A$ in $Cn(M)$. Referring to Example 4-2, set $A = r_8$ such that $r_3$ is also retracted in a contraction. Thus, adding $r_8$ back to the contracted set does not give us the earlier set. Consider the second case where $A$ is true and no other expression in $Cn(M)$ entails it. Then, we have*

$Cn^{+}_{A}(Cn^{-}_{A}(M)) = Cn(\{Cn(M) - \{A\}\} \cup \{A\}) = Cn(Cn(M)) = Cn(M)$.

*4)* $Cn^{*}_{A,B}(M) \neq Cn^{*}_{B,A}(M)$, *unless* $Cn_A(M) = Cn_B(M)$.

As stated in Property 4, the revision of preferences often is an irreversible operation. In addition, a preference set may remain incomplete even through all known preference consequences are deduced. If so, the related agent will prompt its user to enter the missing information, and when even the user is unsure of the answer, it assumes that the expressions are indifferent. This strategy is similar to the principle of indifference in the philosophy of probability [5]: if it is not known that p is more probable than q or that q is more probable than p, then p and q are equally probable.

The indifference principle, sometimes, leads to awkward consequences. Suppose that (i) $p$ is better than $r$, but the ordering relations between $p$ and $q$ and between $r$ and $q$ are undecided. Applying the indifference principle, one infers that (ii) $p$ and $q$ are indifferent and that (iii) $r$ and $q$ are indifferent. The propositions (i), (ii), and (iii) form an awkward triplet – if $p$ is better than $r$, $q$ cannot be indifferent to both, as it would then be better than itself!

This difficulty is notorious and none of the many solutions suggested so far seem satisfactory. Referring to Property 2.2, any undecided preference expression can be asserted into a consistent preference set and the expanded set is still consistent. Property 2.3, however, states that this is not necessarily the case when multiple undecided expressions are added simultaneously. Thus, we restrict an agent to query its user about one undecided preference at a time to sidestep any awkward consequences of the indifference principle. Further, the derived properties of a preference

set also guide us to implement user interface programs that enable domain experts to check the credibility of compiled knowledge and to update preference sets if necessary.

An ordering of preference is formed by sorting the indifference classes of a completed preference set. Section 5 will illustrate some examples on preference orderings in BDN application.

### 4.3 Aggregation

As stated in Section 2, the scheme aggregates individual preferences to provide quick and intuitive answers as feedback to the agents, instead of finding an optimal one using elaborate and difficult mathematics. This subsection briefly describes a standard method of aggregation, the simple majority rule, adopted in BDN. In essence, the simple majority rule takes all criteria to be of equal weight and thus have equal votes on the collective choices. This does not mean that the agents have equal say, however. For any problem, the agenda of that problem may assign an agent to have more criteria than another. Considering two alternatives $x$ and $y$, a group of agents $a$, $b$, and $c$, and the following set of individual preferences regarding criterion $i$, $P(a,i,x,y), P(b,i,x,y)$, and $P(c,i,y,x)$, the aggregated preference of the group using the simple majority rule is $P(i,x,y)$ since two agents favor $x$ over $y$ and one favors $y$ over $x$. Thus an aggregated ordering of preference can be derived from the underlying set of aggregated expressions. In social choice literature, the top ranked alternative(s) of the aggregated ordering are called collective choices.

The simple majority rule also has many useful mathematical properties which are desirable in group decision making. One such property is the Pareto principle, which states that if every individual ordering prefers any $x$ to any $y$, then $x$ is preferred to $y$ in the aggregated ordering. Another property is nondictatorship, which states that if any $x$ is preferred to any $y$, then it must also be the case that $x$ is preferred to $y$ in the aggregated ordering, irrespective of the preference of all other individual orderings.

The simply majority rule may fail to generate an aggregated ordering due to the occurrence of cyclic majorities [25], though such a situation seldom occurs. Consider the preference profile of orderings in Table 1. If alternatives $x$ and $y$ are compared, $x$ is preferred; if $y$ and $z$ are compared, $y$ is preferred; and if $x$ and $z$ are compared, $z$ is preferred. This creates a cycle; no alternative can be the first choice.

The cyclic majorities, that cause the failure of the majority rule, are known as the famous "voting paradox," first observed by Marquis de Condorcet [6, 25]. Since then, there have been many theoretical studies to detect the initial conditions of preferences which cause such a problem. We found, however, that it is computationally more efficient to switch to another aggregation

| Rank | $O_a$ | $O_b$ | $O_c$ |
|---|---|---|---|
| 1 | $x$ | $y$ | $z$ |
| 2 | $y$ | $z$ | $x$ |
| 3 | $z$ | $x$ | $y$ |

Table 1: Cyclic majorities.

method than to check for the initial conditions of cycles. Also, we would like to point out that the computational time of preference aggregation is not an issue here. For non-trivial problems, the generation of initial sets of preference expressions from the underlying knowledge bases would take at least an order of magnitude more.

Hence, instead of making sure that the necessary conditions for cycles found in the social choice literature (e.g., the number of alternatives and the number of criteria in the agenda) will not be present, BDN simply fires the majority rule first, and if any cyclic majorities occur, then the system will automatically resort to another aggregation method: Borda's position rule [3]. This rule assigns weight to any alternative according to its set of positions in the preference profile. After aggregation is completed, BDN then notifies the users of the method used. In addition, the users can select a particular method of aggregation at the beginning of the operation.

Generally, we try to avoid using weighting schemes if possible as we are concerned about the lack of a convenient means for assigning proper weights. Whenever weighting is involved in a decision, there are always disputes over the values of the weights to be assigned. As with other quantitative decision methods, such as probability and utility theory, the disputes can only be settled through lots of experimental data – providing it is possible to obtain such data. In addition, the weights assigned are not guaranteed to hold constant throughout the entire decision-making process and are usually sensitive to the addition of new facts and rules. The earlier attempt on a similar application mentioned in Section 2.1 exemplifies many of these problems.

Since, for non-trivial problems, the knowledge of cooperating agents would be incomplete and inconsistent anyway, the emphasis should not be on the best possible ratings or weights and optimal solutions in the spirit of operation research, but on negotiation for conflict resolution. For this, the presented scheme uses the notion of preferences from social choice theory (both individual and aggregated) as a compact means for multiple agents to express individual viewpoints and as the focal point for the agents to negotiate. The next section discusses the negotiation support of this scheme.

# 5 Negotiation Support

The negotiation of conflicts arising in cooperative decision making often requires human input and intervention. In this scheme, agents rely mostly on human participants to drive the negotiation process and to make final decisions. Based on the formal results of Section 4, we developed many interface facilities to support systematic interaction between the participants and the agents.

In this section, we illustrate some of these interface facilities with representative examples taken from BDN. For purposes of clarity, these examples are concerned only with the preliminary design of a particular structural member of building: beam-column connection. In short, the overall design of a beam-column connection involves four problems, namely, the operation or connection method for moment resistance, om_m; the operation method for shear resistance, om_s; the specific material used for its moment region, dm_m; and the specific material used for its shear region, dm_s. A list of collective choices of these four problems constitutes an overall solution.

## 5.1 Preference Interface

Let us first consider the common display of preference profiles for a BDN agent. Figure 5 exhibits a snapshot captured from the console of the designer agent for a problem, om_m. The left window displays the aggregated ordering at the problem level, such as the preference ordering for om_m shown. The oval icons denote competing alternatives of om_m, and their positions in the ordering are stated in the left column. The lower middle window shows the preference profile of om_m. Note that only reliability and stiffness are the designer's criteria. The information about preference orderings of other criteria in the window are sent by erector and fabricator agents over the network.

Sometimes, more than one alternative may appear in the top rank of an aggregated ordering; this often is caused by insufficient knowledge encoded in the agents for ordering these alternatives. The participants may want to step in and revise the deduced results in order to distinguish the indifferent alternatives. Figure 6 presents an example of the revision of preferences by the designer. This revision is triggered by the difference between alternatives sw_fm_m and fw_sm_m (these refer to welding beam and column at different locations). The designer asserts that fw_sw_m is a better choice than sw_fw_m in regard to the reliability criterion (or issue). Every BDN agent has the authority to change only its own preferences. The new preference ordering for reliability and the aggregated ordering after the revision are shown in this figure. The main window displays system messages from the underlying agent during the revision. Note that the retracted preference expression is an initial one as no other expression is retracted in the process (refer to Strategy 1). The designer can also abort the revision if the result is not satisfactory. In addition, any other

participant in the group can question the participant who initiates a change directly through a mailbox facility.

Through an object-oriented interface package, the graphical objects displayed directly correspond to the objects being represented in the underlying knowledge base. One can query and modify the knowledge base through graphical objects. In Figure 7, for instance, the designer queries the object beam about its competing alternatives for problem dm_s and finds five alternatives. (Items at the top portion of the window are the alternatives selected for the connection.) Such information may not necessarily reside in the local knowledge base. Nevertheless, the underlying knowledge server (local communication knowledge module) will automatically retrieve the information from other knowledge bases. Such retrieval is transparent to the human participants. The basic paradigm for such an object-oriented interface package has been presented in [32].

## 5.2 Knowledge Retrieval

The preference scheme adopts the following strategy of negotiation for BDN application.

**Strategy 2** *(Negotiation of preferences)*

*1) Each of the agents computes a negotiation index by calculating the ranking difference of every feasible alternative between its individual ordering and the aggregated ordering.*

*2) Each of the agents checks if its index is over a threshold value, and if so, asks its user whether to complain or not; otherwise, the agent exits.*

*3) If any of the agents complains, bargaining is started, and if bargaining fails, forcing is attempted.*

*4) If there is a change of preferences, the aggregated ordering and collective choice are recomputed, and the process returns to step 1. Otherwise, a consensus is reached, and the process is exited.*

In Step 1 of the strategy, every agent first obtains a heuristic index of negotiation. Consider the simple case in Table 2. Agent $b$ calculates $x$'s ranking difference between its ordering $O_b$ and the aggregated ordering $O_g$ as 1 and the total difference, that is, its heuristic index $H_b$ as 4. Similarly, we have $H_a = 0$ and $H_c = 2$. An agent uses its heuristic index to decide whether to flag the users for conflicts and to estimate whether the negotiation is converging towards a satisfactory solution. In Step 2, suppose that a uniform threshold, $H_{th} = number\ of\ alternatives - 1 = 2$, is applied

| $O_a$ | $O_b$ | $O_c$ | $O_g$ |
|---|---|---|---|
| $z$ | $x$ | $z$ | $z$ |
| $x$ | $y$ | $y$ | $x$ |
| $y$ | $z$ | $x$ | $y$ |

Table 2: Individual and aggregated orderings.

across all agents. (Such an index is derived empirically and is for a specific aggregation method.) Thus, the user of $b$ would like to complain about the outcome as $z$ is its least preferable choice.

As mentioned in Section 2.3, the predominant mode of resolving such a conflict in this scheme is the use of bargaining or compromise. In [31], we have described many high-level communication protocols which use the logic of illocutionary acts [23] as a formal basis for orderly interactions among agents during bargaining. One distinct feature of the bargaining in this scheme is that the participants can probe into the background knowledge of the agents (both the local and remote agents) which deduces conflicting preferences. This facility enables the participants to better understand the intention and perspectives of one another. In what follows, we describe the knowledge structure that supports such a retrieval.

For simplicity, let us consider that the classical representation of production or inference rules for knowledge bases [9]:

$$p_1 \wedge p_2 \wedge \cdots \wedge p_n \supset q$$

where $p_1, \cdots, p_n$ are premises and $q$ is a conclusion based upon these premises. In addition, only a conjunction of premises is allowed, since it is always possible to represent the disjunction with the help of at least two equivalent rules. The rules have only one element on their right-hand side.

Example 5-1

1) The rule, $p_1 \wedge p_2 \vee p_3 \supset q$, is equivalent to the pair of rules: $(p_1 \wedge p_2) \supset q$; $p_3 \supset q$.

2) The rule, $p_1 \wedge p_2 \supset q_1 \vee q_2$, can be represented as either $p_1 \wedge p_2 \wedge \neg q_2 \supset q_1$ or $p_1 \wedge p_2 \wedge \neg q_1 \supset q_2$.

3) The rule, $p_1 \wedge p_2 \supset q_2 \wedge q_1$, is equivalent to the pair of rules: $p_1 \wedge p_2 \supset q_1$; $q_1 \supset q_2$.

It is worth noting that such a rule (with a single element conclusion and a conjunction of premises) is equivalent to a Horn clause: $\neg p_1 \vee \neg p_2 \vee \cdots \vee \neg p_n \vee q$. This motives the use of the logic programming language for our implementation.

The symbolic structure of a premise is more complex in the BDN system. It can be a logical predicate, an attribute of a knowledge base object, or a meta-predicate [30, 29]. It is possible to build an AND/OR graph based on linking rules such that the action of the first is a premise for the second. The goals (leaf nodes) of this graph then form a set of initial preference expressions. Since preference constraints are also represented in the production rules, they can be added to the graph. Such a graphical structure of knowledge would explicitly capture the reasoning steps of individual agents.

For logic-programming based systems, the dynamical tracing of reasoning steps can be achieved easily with the *why* meta-interpreter [4, 27]. In BDN, such an interpreter is invoked by the *why* option in the local preference profile window or by a *why* message of a remote agent. In the current BDN system, incremental modification of background knowledge during negotiation is possible but is monitored by the human participants. Further, discussions by participants concerning all modifications is recorded.

As an example, Figure 8 shows that the designer queries the fabricator agent as to why the two alternatives, $tee\_m$ and $angle\_m$ (tee and angle), are indifferent in regards to the fabrication cost of materials used for moment-resistant connection. The fabricator agent traces back the immediate justifications: $fab\_cost\_index(angle\_m, 1.25)$ and $fab\_cost\_index(tee\_m, 1.25)$ and sends them to the designer agent together with the textual explanation of the fired rule, that is, $fab\_cost\_index(angle\_m, 1.25) \wedge fab\_cost\_index(tee_m, 1.25) \supset I(angle\_m, tee\_m)$.

The designer agent then displays the information in its explanation window. In this case, both alternatives have the same fabrication cost index. Not satisfied with the answer, the designer asks the fabricator agent further why the alternative $angle\_m$ has a cost index of 1.25. A sequence explanations of repeated *why* queries about the indifference expression would then be followed. The trace stops when repeated queries lead to primitive fact(s) or data of the fabricator agent's knowledge base.

## 6 Conclusion

Recent advances in computer technologies have opened up new vistas for intelligent interaction among several AI systems for solving larger, more complex problems. In this paper, we have described a new problem solving scheme for cooperative knowledge-based systems that uses social choice theory as a formal basis for making joint decisions and promoting conflicts resolution. We discussed the three operational steps of this scheme, namely, identification, processing, and negotiation (Section 2), and presented some of the scheme's formal properties (Sections 3 and

4). The distinct features of this scheme are that: (1) it is qualitative, interactive, and largely deductive in nature, and (2) it uses social choice functions only as 'preference revealers' to support negotiation among agents to reach consensus (Section 5), rather than to calculate 'optimal' solutions.

This research is application-driven. The formal scheme has been used in developing a working CKBS prototype, BDN, for preliminary building design. The practicality of the scheme is also illustrated with real-world examples from BDN. The main message of this paper is that decision making in CKBS requires a paradigm different from those based on conventional game theory or operational research [28, 14, 19]. For non-trivial problems, credible, well-behaved quantitative data about competing alternatives is seldom obtainable or consistent among agents. Moreover, such data is unlikely to be applicable as knowledge bases evolve. An earlier attempt using cardinal ratings for a similar application exemplifies many of the shortcomings of the conventional paradigm (Section 2.1).

For cooperative decision making, the real issue is not whether qualitative or quantitative data works better, but whether the model of decision making is *logical*. A logical model is capable to derive the decision data systematically, to explain how or why preferences arise, to provide the chain of reasoning behind them, and to allow the modification of them when necessary. In other words, one should avoid the *black box* approach when building cooperative decision-support knowledge systems. The purpose of such systems is to assist the users to make better decisions, not to dictate the solutions to decision problems.

This preference scheme provides such a logical model to represent the decision making process. It adopts binary expressions simply because the background knowledge of these qualitative expressions is generally easier to acquire than that of quantitative ones such as cardinal ratings and certainty factors.

The use of preference expressions also causes certain uncertainty in the aggregation (Sections 2.2 and 4.3). This, however, is not as severe as it looks. The object of this scheme is not to formulate 'optimal' solutions as in the conventional paradigm but to use the results of aggregation as the focal point for negotiation. The negotiation among agents aims to iron out the differences and uncertainties of individual preferences in order to reach a consensus. In addition, we found that the users are more susceptible to the collective choices made in such an interactive reasoning system than the 'optimal' solutions derived through non-interactive means.

Finally, two areas for further research are identified. First, for many problems such as building design, there is no *a priori* logical or philosophical principle on which to base a resolution strategy

– *correct* resolution depends on specific domain knowledge as well as the cooperative behavior of the participants. The resolution of conflicts in the current scheme is largely driven by the participants, with preference profiles and collective choices as focal points. Future work will be to understand more of the human reasoning steps involved and to incorporate them into the scheme.

The second related issue is the revision of background knowledge and heuristics. Many rule-based revision systems have been developed to update the knowledge-based systems that change their states over time [21, 10, 11]. Most revision systems, however, concerns the syntactical aspects of the revision, i.e., the consistency criteria defined by certain nonmonotonic logic representation of knowledge. For problems of cooperative nature, such as building design, the maintenance of the semantics, such as design intentions among agents, is a critical issue. The BDN system relies partly on the participants checking the semantics of the revision, and this, sometimes, distracts the participants from resolving conflicts. Further work will be to study revision procedures for the maintenance of design intentions.

## 7 Acknowledgement

This work is based on doctoral research done by the author at the NSF Engineering Research Center: Advanced Technology for Large Structural Systems (ATLSS) at Lehigh University, Bethlehem, PA 18015. The author would like to thank John Wilson, Donald Hillman, and ICOT researchers of knowledge-based management systems group for many valuable discussions. The author would also like to acknowledge the support of ICOT in writing and presenting this work.

## References

[1] Arrow, K. J. *Social choice and individual values*, New York, Wiley, 1951.

[2] ATLSS Center. *Sixth-year renewal proposal to the NSF, Vol. 2: Projects, Publications, and Biosketches*, ATLSS Drive, Imbt Lab., Lehigh University, Bethlehem, PA, 1992, Section 7.

[3] Borda, J. C. *Memoire sur les Elections au Scrutin*, 1781, English translation by A. De Grazia, *Isis*, vol. 44, 1953.

[4] Bratko, I. *PROLOG - Programming for artificial intelligence*, 2nd edition, Addison Wesley, 1990, Chapter 20.

[5] Carnap, R. *Logical foundations of probability*, Chicago, University of Chicago Press, 1962.

[6] Condorcet, M. *Essai sur l'Application de l'Analyse a la Probabilitie des Decisions rendues a la pluralite des voix*, Paris, L'Imprimerie Royale, 1785.

[7] Corbin, J. R. *The art of distributed applications*, Springer-Verlag, New York, 1990.

[8] Davis, M. *Game theory*, New York, Basic Books, 1973.

[9] Davis, R., Buchanan G., Shortliffe E. H., Production rules as a representation of a knowledge base consultation system, *Artificial Intelligence*, vol. 8, 1977, pp. 15-45.

[10] de Kleer, J. An assumption-based TMS, *Artificial Intelligence*, vol. 28, no.2, 1986, pp. 127-162.

[11] Doyle, J. A truth maintenance system, *Artificial Intelligence*, vol. 12, no.2, 1979, pp. 231-272.

[12] Durfee, E. H., Lesser, V. R., Corkill, D. D. Cooperative distributed problem solving, in Barr, A., Cohen, P. R., Feigenbaum E. A. (eds.), *The handbook of artificial intelligence*, vol. iv, (1989), 83-148.

[13] Ebbinghaus, H. *Memory*, New York, Wiley, 1885; translated in 1964.

[14] Ephrati, E., Rosenschein, J. S., The clarke tax as a consensus mechanism among automated agents, in *Proceedings 9th AAAI Conference*, July, 1991, pp.173-178.

[15] Gardenfors, P. *Knowledge in flux - modeling the dynamics of epistemic states*, MIT press, Cambridge, Massachusetts, 1988.

[16] Gasser, L., Hill, R. W. Coordinated problem solvers, in *Annual Reviews of Computer Science*, vol.4, (1989-90), pp. 203-254.

[17] Galbraith, J. *Designing complex organizations*, Reading, Mass., Addison-Wesley, 1973.

[18] Galbraith, J. *Organizational design*, Reading, Mass., Addison-Wesley, 1977.

[19] Kraus, S., Wilkenfeld, J., Negotiations over time in a multi-agent environment: preliminary report, in *Proceedings 12th IJCAI Conference*, Sydney, Australia, August, 1991, pp.56-61.

[20] Malone, T. W. Organizing information processing systems: parallel between human organizations and computer systems, in Robertson, S. P., Zachary, W. W., Black J. B., (eds.) *Introduction to Cognition, Computation, and Cooperation*, Ablex, 1990, pp. 56-83.

[21] Morizet-Mahoudeax, P. Maintaining consistency of a database during monitoring of an evolving process by a knowledge-based systems, *IEEE Transactions on systems, man, and cybernetics*, vol. 21, No. 1, 1991, pp.47-60.

[22] Pearle, J. *Probabilistic reasoning in intelligent systems*, Morgan Kaufmann, 1988.

[23] Searle, J. R., Vanderveken D. *Foundation of illocutionary logic*, Cambridge University Press, 1985.

[24] Sen, A. *Collective choice and social welfare*, San Francisco, Holden-Day, 1970.

[25] Sen, A. *Choice, welfare, and measurement*, Cambridge, Mass., MIT press, 1982.

[26] Shafer, G. *A mathematical theory of evidence*, Princeton, Princeton University Press, 1976.

[27] Sterling, L., Yalcinalp, L. U. Explaining Prolog based expert systems using a layered meta-interpreter, In *Proceedings 11th IJCAI*, 1989, pp. 66-71.

[28] Sykes, E. A., White, C. C., Multiobjective intelligent computer-aided design, *IEEE Trans. on Systems, Man, and Cybernetics*, Nov/Dec., 1991, pp. 1498-1511.

[29] Wong, S. T. C. *A framework for description and design of cooperative knowledge-based systems*, Ph.D. thesis, CSEE Department, Lehigh University, Bethlehem, PA, Oct. 1991.

[30] Wong, S. T. C., Wilson, J. L. A set of design guidelines for object-oriented deductive systems, *IEEE Trans. on Data and Knowledge Engineering*, (to appear) June, 1993.

[31] Wong, S. T. C. A communication scheme for cooperative systems communication, in *Proceedings of Multi-agents Cooperative Computing Workshop*, Kobe, Japan, December, 1992.

[32] Wong, S. T. C., Wilson, J. L. Extending knowledge-based systems through closely-coupled graphics and windows, in *Proceedings of the second international conference on industrial & engineering applications of small AI and expert systems*, Tullahoma, TN, June, 1989.

[33] Von Neumann, J., Morgenstern, O. *Theory of games and economic behavior*, Princeton, Princeton University Press, 1944.

[34] Zadeh, L. A., Fuzzy logic and approximate reasoning, *Synthese*, Vol. 30, 1975.

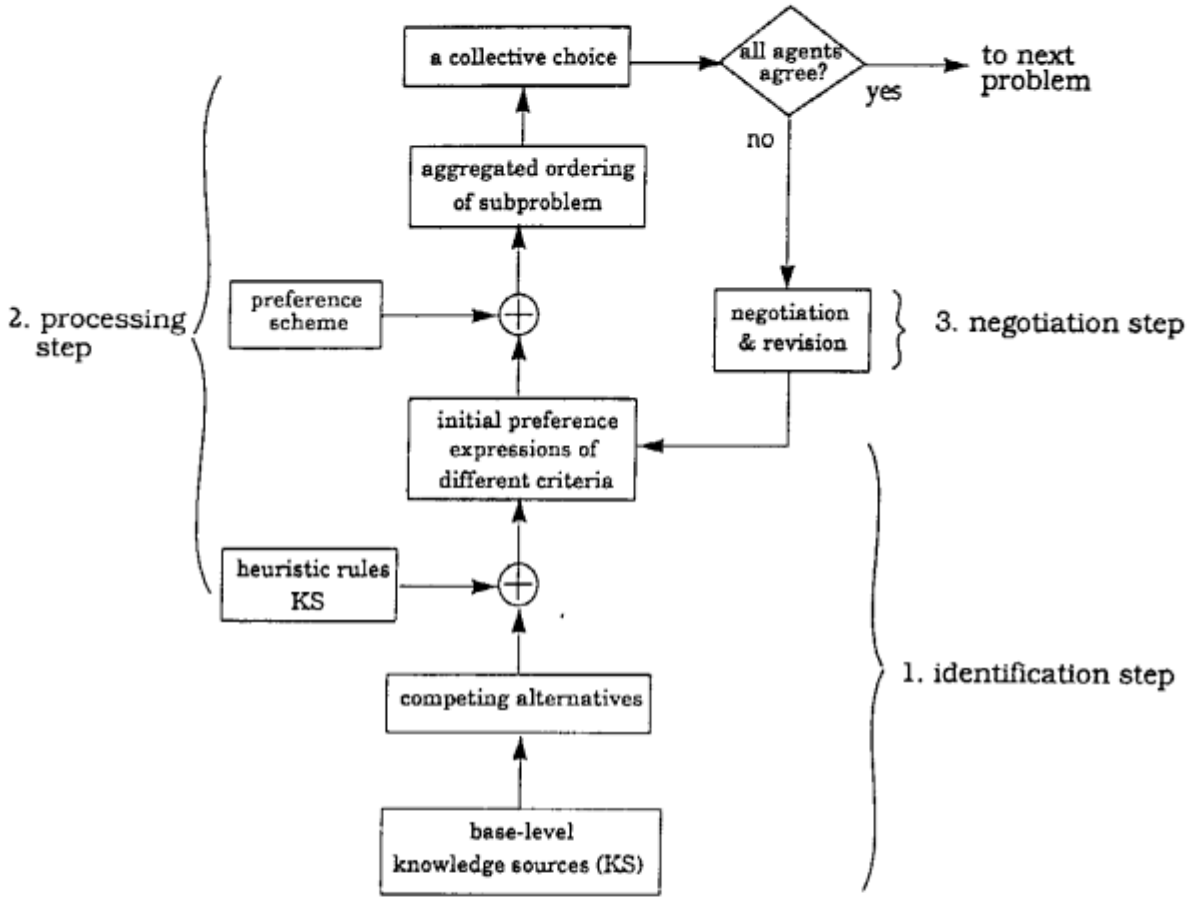


Figure 1. The three steps of the scheme for cooperative decision-making.

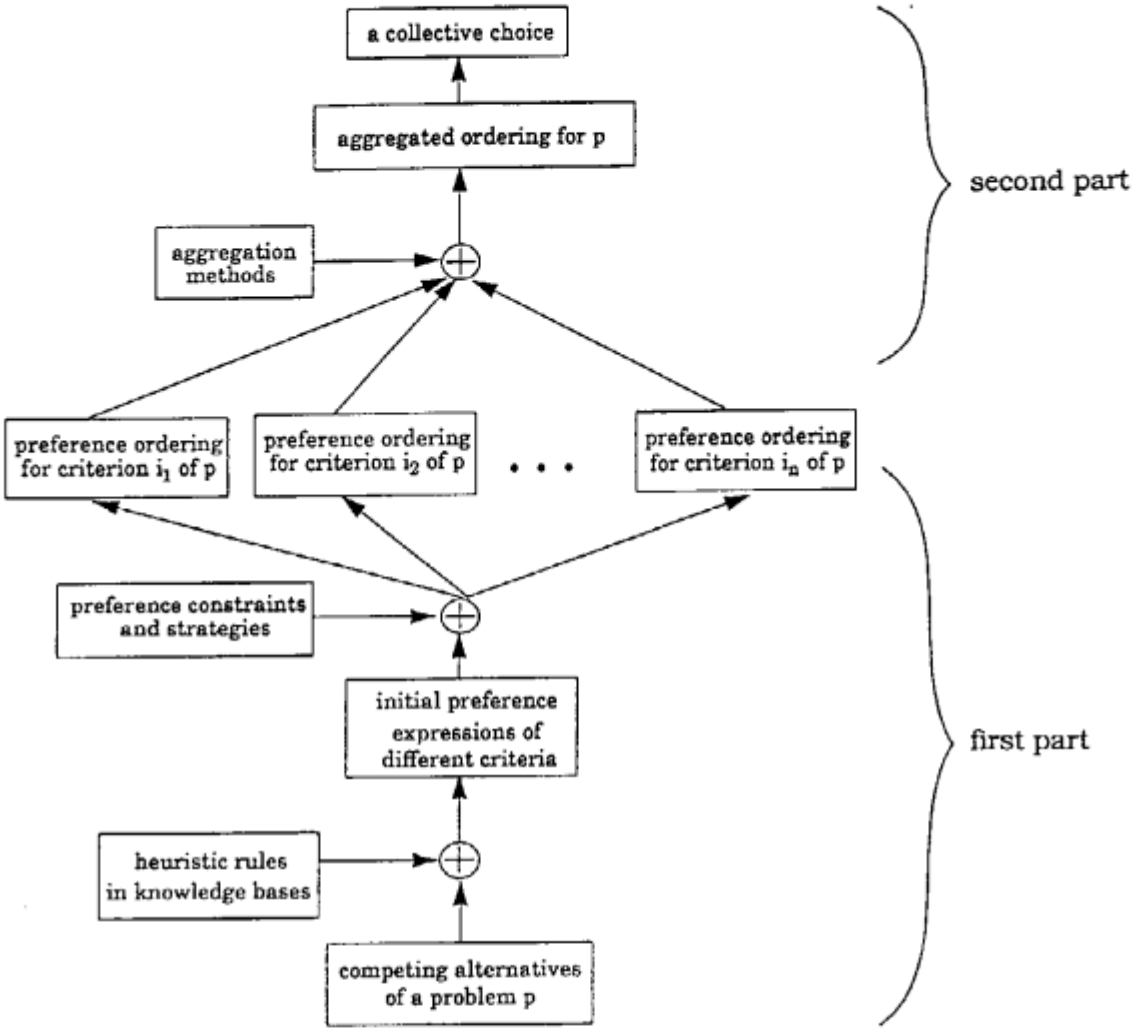


Figure 2. The information flow diagram of the processing step of the scheme.


Figure 3. A graphical illustration of ordering relations between $x$, $y$, and $z$.

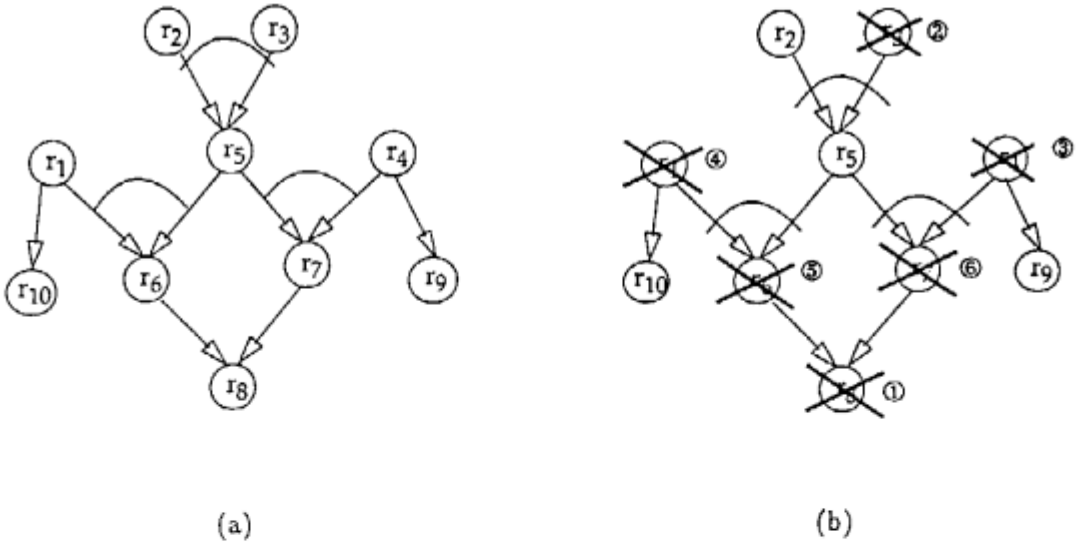


Figure 4. Precedence relation graphs.

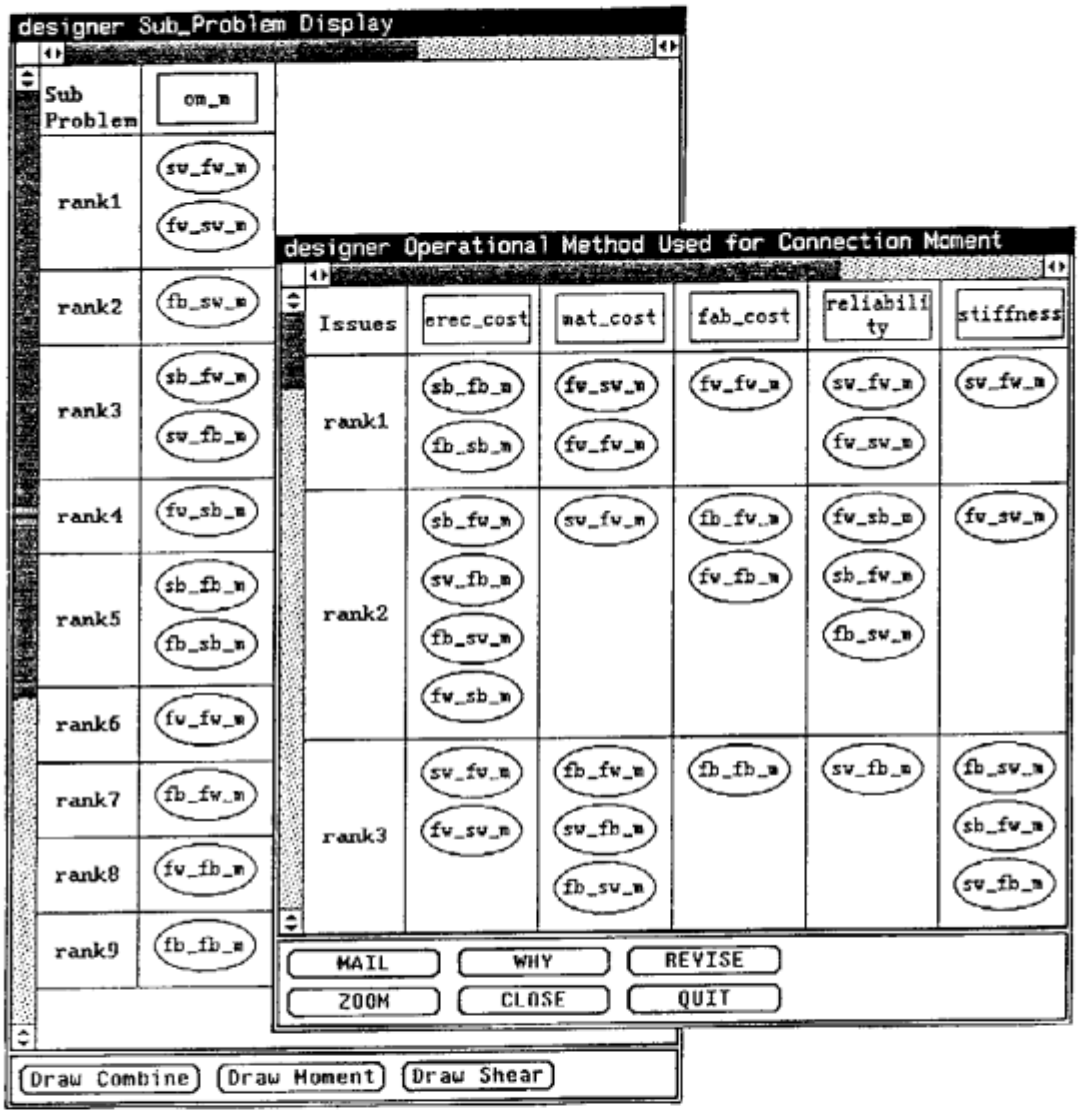


Figure 5. A snapshot of the designer's preference profile and aggregated ordering for *om_m*.

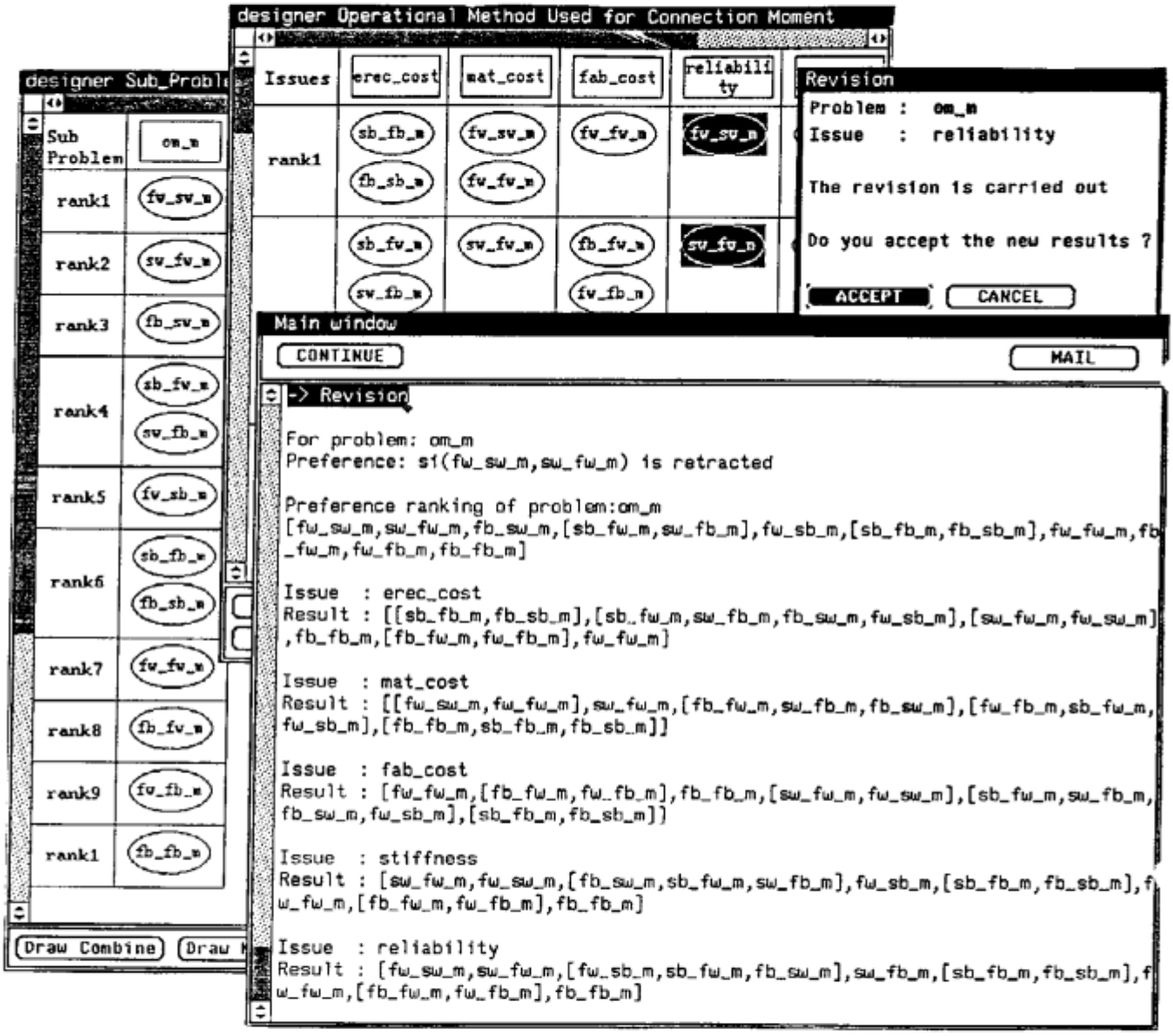


Figure 6. A snapshot view of a revision process.

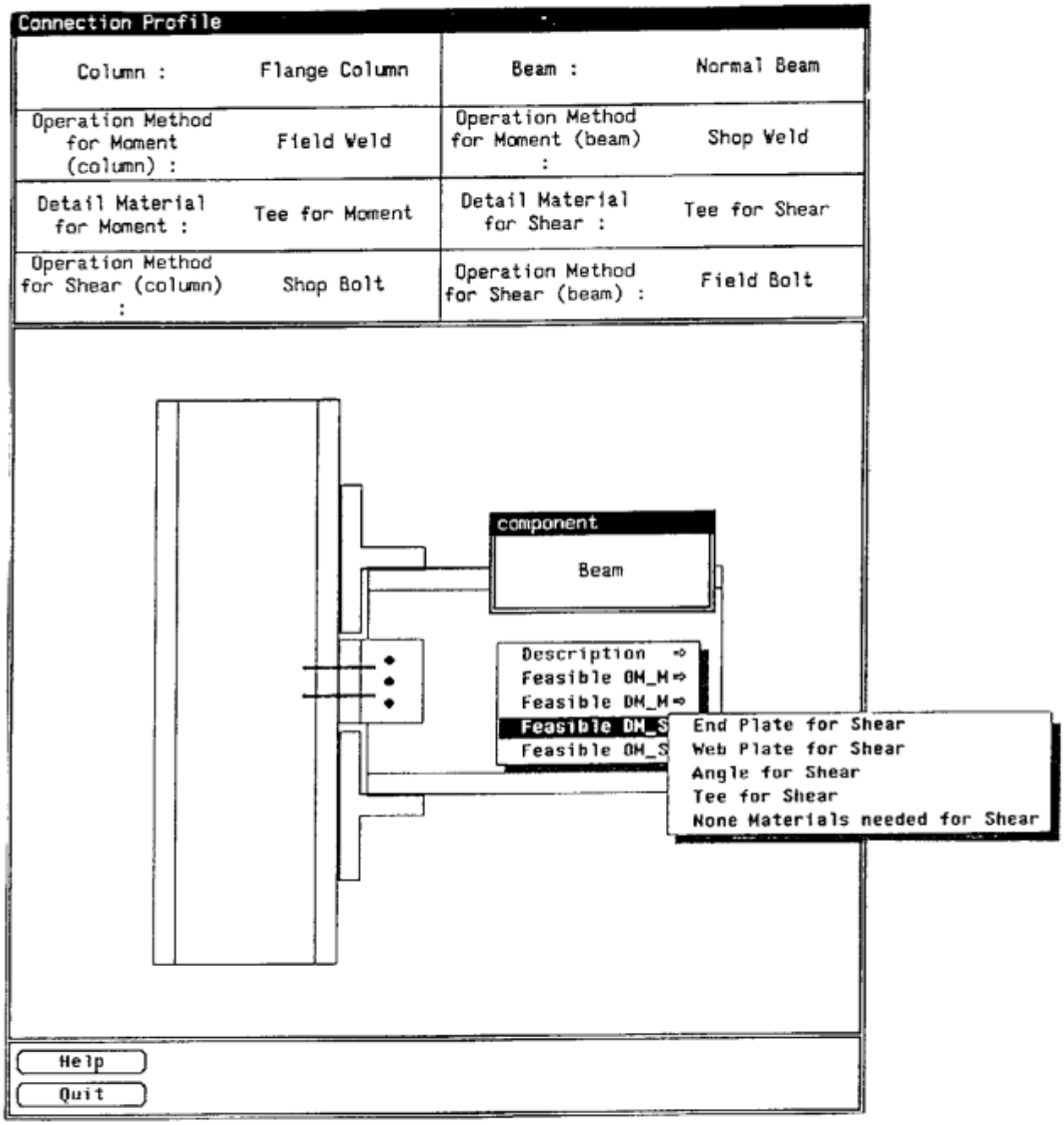


Figure 7. Pictorial query of information residing in knowledge bases.

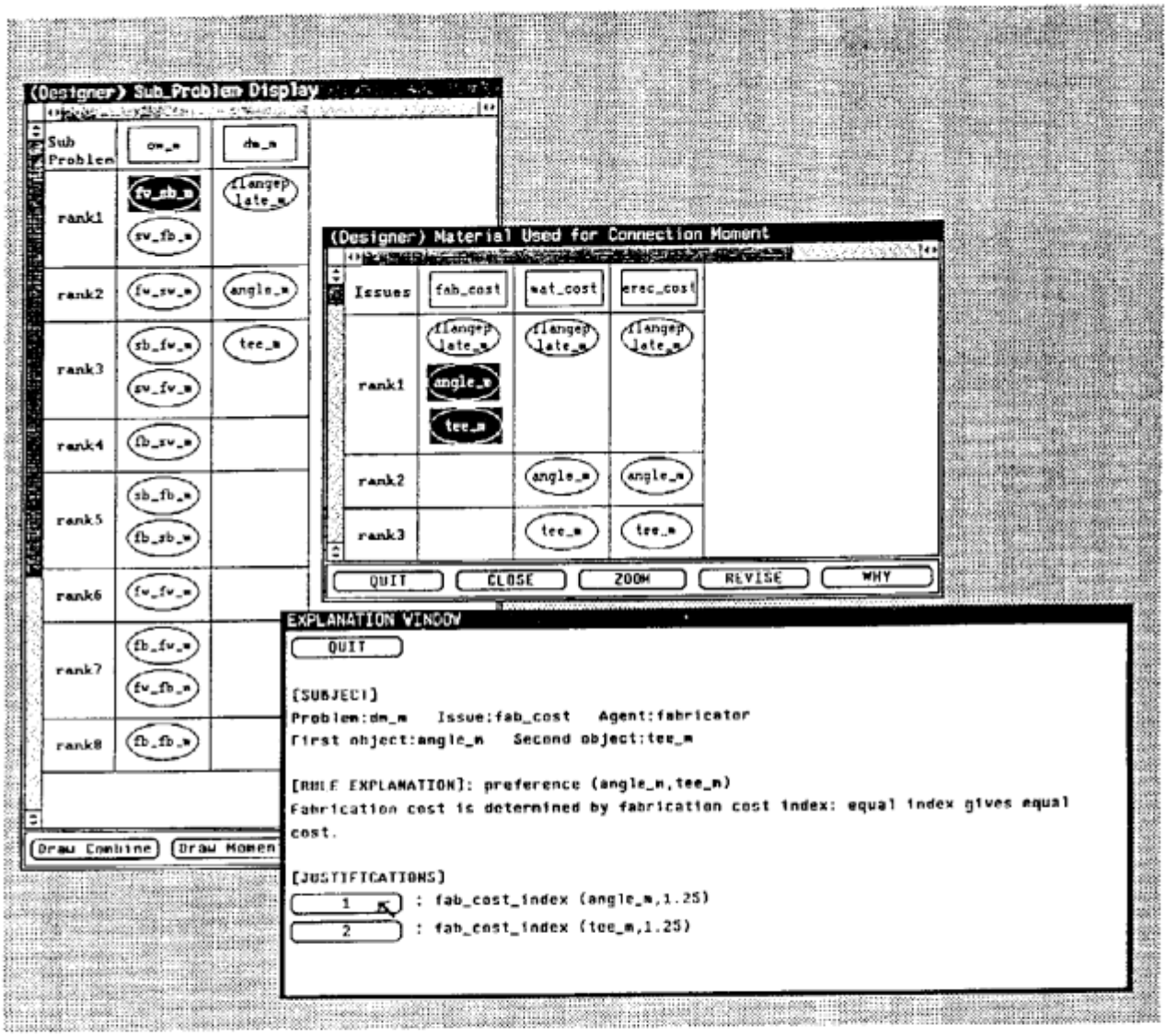


Figure 8. Explanation of the indifference of *tee_m* and *angle_m* in subproblem *dm_m*.