Barbuceanu, M., and Fox, M.S., (1997), "Conflict Management with a Credibility/Deniability Model", Working Paper, Enterprise Integration Laboratory, University of Toronto.

Conflict Management with a Credibility/ Deniability Model

Mihai Barbuceanu and **Mark S. Fox**

Enterprise Integration Laboratory
Department of Industrial Engineering
University of Toronto
4 Taddle Creek Road, Rosebrugh Building, Toronto, Ontario, M5S 1A4
email: {mihai, msf}@ie.utoronto.ca, tel: 1-416-978-0910, fax:1-416-978-3453

Abstract

When a reasoning system encounters a contradiction like p&q=>false, it may try to eliminate it by retracting some belief that supports either p or q. This paper addresses the issue of how to determine which of the possible supporting beliefs to retract. The problem is studied in the context of organizational multi-agent systems in which we can not make assumptions about how agents derive their beliefs. The presented model considers two properties of beliefs: the credibility (competence) of the agents that provided the belief in the first place and the costs incurred upon other agents that already used the belief (directly or indirectly) for their own decision making or action, if the belief is to be retracted. The model is implemented as a service of an agent programming shell we have created for enterprise integration.

Keywords: credibility, retraction cost, conflict management, cooperative information systems, enterprise integration.

1.0 Introduction

When a reasoning system encounters a contradiction like *p&q=>false*, it may try to eliminate it by retracting some current belief that supports either *p* or *q*. This paper addresses the issue of how to determine which of the possible supporting beliefs to retract. The problem is studied in the context of organizational multi-agent systems in which we can not make any assumptions about how agents derive their beliefs.

The model considers two properties of beliefs: the *credibility* (competence) of the agents that provided the belief in the first place - as ascribed to them by the organization they are part of - and the *costs* incurred upon other agents in the organization that used the belief for their own decision making or action, if the belief is to be

retracted. Credibility is organizationally defined by a partial order of agents in different organizational roles. The model is implemented as a service of an agent programming shell we have created for building collaborative information systems for enterprise integration.

We consider three main advantages of this model. First, it provides a general method for belief revision that is accurate, as it depends on the current views of all involved parties, rather than on a pre-defined scheme. Second, the model allows reasoning about conflict management and belief retraction based on domain knowledge. Third, it minimizes the communication and negotiation overhead by identifying situations in which negotiation can be avoided or reduced.

The paper presents first the agent-based information architecture that forms the context of our work. Then it presents the organization ontology that we use to define agent credibility. Finally, it discusses the proposed credibility/deniability model and reviews a number of issues that occured in the implementation phase.

2.0 The Generic Agent Shell

Our research approaches the construction of collaborative enterprise information architectures [18, 20] by adopting an agent-oriented view. We are developing a generic agent programming shell that will be used to build multi-agent architectures. This approach ensures the ability to reuse abstract descriptions of system components, services, knowledge bases and coordination models. Amongst the services provided by the shell is the conflict management one, forming the subject of this paper.

Before describing this model in detail, we review in this section the architecture and the major services provided by the Generic Agent shell.

2.1 Architecture of the Generic Agent

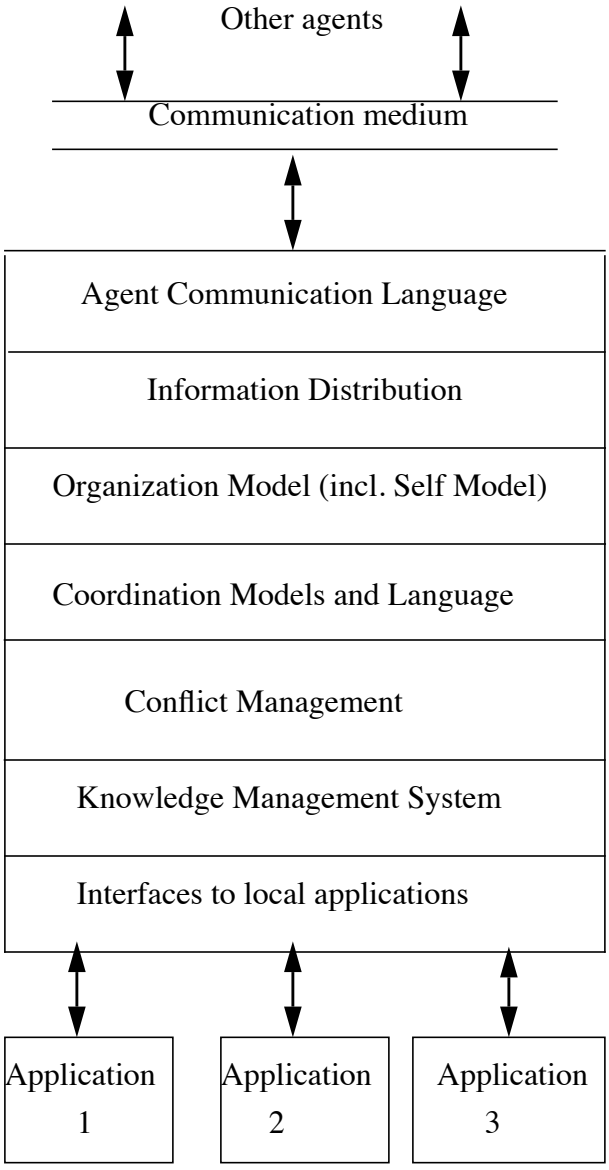


FIGURE 1. Architecture of the Generic Agent

The Generic Agent shell is composed of several layers of languages and services, as shown in figure 1:

1. *Agent Communication Language*. This is the language agents use to communicate. The language consists of speech acts, describing the communicative actions carried out, and an embedded content layer that describes the actual information communicated. We have adopted as the ACL the KQML/KIF [10, 13] language produced as a result of the ARPA effort on knowledge sharing. KQML/KIF supports a declarative approach to knowledge communication, as opposed to procedural approaches (e.g. TeleScript). A major advantage of the declarative approach is that it permits the explicit and declarative specification of various coordination mechanisms agents use. These coordination mechanisms can be modeled as shared conventions governing the exchanged speech acts during coordinated action. These shared conventions allow participating agents to understand the intentions of the other agents and thus to coordinate effectively in solving the common problem.

2. *Information distribution*. The information distribution service is a generic whose major purpose is to be able to distribute (voluntary or at request) information of interest to other agents, in a manner that relies on the content of the information. This essentially requires deductive information processing capabilities. Another capability of the service is performing multi-agent belief revision functions. When any knowledge or information used by the agent is invalidated, the service determines if and what communicated information is invalidated and sends denial messages to the recipients of that information.

3. *Organizational model*. Agents can not operate autonomously unless they have a model of the organization(s) they are part of. This model tells the agent what other agents exist, which are the roles they have, what goals agents pursue, what sort of communication can take place amongst them, etc. The organization model contains a model of the agent itself, representing the agent's roles, goals and capabilities.

4. *Coordination models and language*. Coordination models - shared conventions about exchanged messages during cooperative action - are described in a special purpose coordination language. We are building on the assumption that coordination models can be generically defined in terms of rules about cooperation and situation assessment that are applicable in most (if not all) industrial applications.

The purpose of the coordination language is to allow the explicit, declarative specification of coordination models. Some mechanisms will be more general, others may be more application specific. The coordination language allows the representation of coordination models in coordination libraries that are imported and extended by applications. The implemented system has a graphic user interface allowing users to manipulate visual representations of coordination models.

5. *Conflict management*. The conflict management service enables the agent to make decisions when confronted with contradictory information derived or received from other agents. We assume that in real enterprises contradictory information occurs quite often and being able to cope with it increases the robustness of agents. We distinguish among several levels of consistency:

- *Terminological consistency* refers to the coherence of the conceptual vocabulary employed by agents and is handled by the T-Box services of the underlying description logic used by the IA.
- *Assertional consistency* refers to the coherence of the infornation exchanged by agents and is handled by the mechanisms presented in this paper.
- *Temporal consistency* as the coherence of exchanged information wrt to time intervals during which beliefs are held. The approach presented here handles this as well, as will be shown later on.

6. *Generic interface to applications*. An agent may control a number of non-agent applications (legacy or purpose built). For this purpose it must provide an interface allowing data, parameters and control specifications to be transmitted to/from the applications. We are assuming that even if the integrated applications will be diverse, the interface need not be ad-hoc. In consequence, we have devised ways to construct a systematic interface able to accommodate diversity at one end and consistent manipulation mechanisms at the other.

2.2 The Knowledge Management System

Agents need to store and process knowledge locally. Services like content based information distribution impose requirements on how local knowledge is to be processed (for example, deductive inference is needed for content based distribution, truth-maintenance type capabilities are needed for belief revision, etc.). Other requirements come from the enterprise modeling domain, such as the ability to represent complex enterprise models and to reason about common-sense notions like time. For these reasons, we are providing the agent shell with a powerful description logic representation of knowledge extended with temporal reasoning and several other processing mechanisms.

We use a description logic language [3] that provides the usual concept-forming operators - *conjunction*, *value restrictions*, *number restrictions* - roles and subroles, disjointness declarations, primitive and defined concept specifications. The language T-Box provides the usual services of constructing the *complete form* of concepts and *automated classification* based on *subsumption checking* [5, 6, 14, etc.] The language A-Box is essentially a *constraint propagation* engine that makes instances conform to the various constraints asserted about them. It uses a propositional representation of instances and roles.

Unlike usual A-Boxes, ours is also a full temporal reasoning system that (i) records the *time intervals* during which propositions are true or false [1], (ii) provides a *query language* that supports temporal interrogations, (iii) provides for the definition and application of *rules that perform forward reasoning* extending the information in the time mapped data base and (iiii) provides a *temporally extended* boolean truth maintenance system that records dependencies and supports asserting and retracting time-mapped propositions.

3.0 Organizational modeling

Agents can not operate autonomously unless they have an understanding of the environment they are in. This understanding consists of models of the other agents in the environment, the roles they play, the goals they are pursuing, the actions they are empowered to execute, the

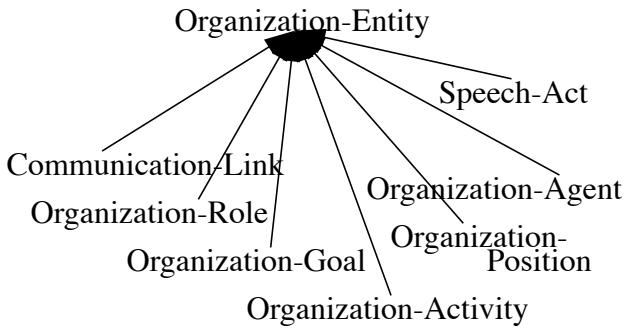
information they are interested in, the services they can provide, the established communication and authority channels, etc. [12, 17].

To endow our actual agents with this capability we have developed an organization ontology that provides the necessary distinctions - terms and relations - for describing organizations and agents. For the purposes of this paper, the organization ontology is used to define the notion of agent credibility the conflict mangement model relies on.

3.1 Basic elements

Figure 2 shows the basic elements of our organization ontology.

FIGURE 2. Organizational entity taxonomy



Organization-Agents (OAs) are the "active" entities in an organization. They represent either individuals, like employees and contractors, or groups like departments, divisions, boards of directors, etc. (figure 3).

OAs play various Organization-Roles, they have Organization-Goals to achieve, fill Organization-Positions, communicate with other OA using Communication-Links and Speech-Acts. The following subsections detail the meaning of these concepts.

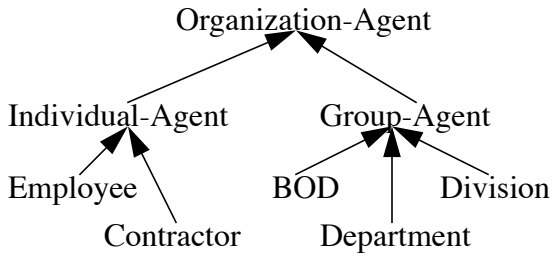


FIGURE 3. Organization-Agents

3.2 Organization-Role

An Organization-Role defines a prototypical function of an agent in an organization. A particular agent can assume several roles in the same time. For an individual agent, examples of organizational roles include "project supervision", "customer liaison", "system administration", "C++ guru", etc. Once an agent is assigned to a role, that creates a commitment on the agent's part to act in order to achieve the goal(s) of the role.

Each Organizational-Role has:

- *Goals*: one or several Organization-Goals the agent playing the role is responsible for.
- *Skills*: one or more skills required to achieve the goals
- *Processes*: activity networks performed to achieve the goals
- *Policies*: constraints on the performance of the role's processes. These constraints are unique to the organization role.
- *Communication-Link*: these are communication links to other agents in specified roles. Communication consists of exchanging speech acts according to specific conversation structures that are also formally represented.

3.3 Organization Position

An organization position defines a formal position that can be filled by an OA in the organization. Examples of positions include “professor” , “laboratory director”, “senior researcher”, “sales-representative”, etc. Any position essentially consists of a set of roles the OA filling it will have to carry out. For each positions we specify:

- *Roles*: the roles to be assumed in the position
- *Filling-Agent*: the organization agent filling the position. In general we assume that positions are filled by individual agents.
- *Policies*: constraints on the performance of position’s processes (inherited from the required roles). These constraints are unique to the organization position.

3.4 Organization Goals

An organization can be seen as a mechanism for decomposing goals into subgoals and achieving these subgoals by assigning them to organization agents. Therefore the representation of organization goals is important as it captures the reasons why an organization exists in the first place. The representation of organization goals includes:

- *Conjunctive-goals*: a decomposition into a set of goals such that all goals must be achieved
- *Disjunctive-goals*: a decomposition into a set of goals such that at least one of the goals must be achieved
- *Depends-on*: dependencies among goals: a goal G1 depends on a goal G2 if G2 must be achieved before G1 can be achieved.

3.5 Organization Activities

Organization-Roles associate Organization-Goals with activity networks used to achieve the goals. We distinguish among two major kinds of activities (figure 4).

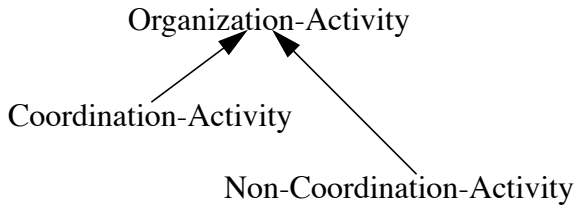


FIGURE 4. Organization activities

In the class of Non-Coordination-Activity we have a general representation of activities that is used for management, manufacturing and other activities agents perform. Coordination-Activities describe the structure of coordinating conversations among the current agent and other agents carried out during cooperation.

In each role, agents have a goal to achieve. This goal may have conjunctive or disjunctive subgoals. Leaf subgoals have associated activities that can be carried out to achieve the subgoal. If the activity is a Non-Coordination-Activity, it can be carried out locally by the agent itself. If the activity is a Coordination-Activity, it implies cooperation with other agents and is executed by exchanging messages with other agents. Messages consist of KQML speech acts. The structure of the negotiation protocols that govern message exchange for Coordination-Activities is discussed elsewhere [4].

3.6 Communication Links

We distinguish among two kinds of communication links, the Information-Distribution link and

the Communication-With-Commitment link (figure 5).

3.6.1 Information-Distribution

The Information-Distribution link is used by agents who perform cooperative information distribution functions. It describes, for an agent in a given organizational role, the information it is interested in and the information it can supply to other agents.

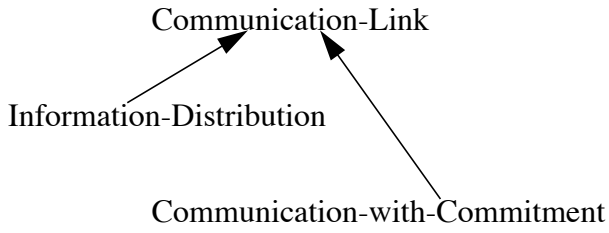


FIGURE 5. Communication links

The Information-Distribution link specifies:

- *Agent*: the agent connected to the link
- *Role*: the organization role played by the agent
- *Interests*: the information interests of the agent
- *Volunteers*: the information it can supply to other agents

It is understood that information distribution in the above case is uncommital, in the sense that it does not create obligations for either the sender or the receiver.

3.6.2 Communication-with-Commitment

The second type of communication does create commitments. A Communication-with-Commitment link is a *unidirectional* channel between a Committing-Agent and a Committed-Agent, each in specified roles. The communication consists of sending speech-acts between the two connected agents - according to whatever protocol is selected by the agents - with the committed agent being obliged to carry out in good faith whatever actions are implied by the communication.

This is a way of defining the *authority of action* (or simpler, authority) of the Committing-Agent in the given role. Agents have different authorities depending on the roles they are in. For example, if the Committing-Agent is an agent in a Plant-Management role and the Committed-Agent is an agent in a Resource-Allocation role, the former can send the latter a performative like:

```
(use resource-007 activity-09).
```

This means that the sender has authority to commit the receiver into using **`resource-007`** to **`activity-09.`**

This notion of authority covers many necessary meanings of the term, such as:

- *Authority to allocate resources*. Authority may be restricted in that how much resource, or at what time or to which activities the resources are allocated can be delimited by a higher authority.

  An agent may have authority over any type of resource, including other agents. This implies that an agent may assign others to an activity.
- *Authority to perform an activity*. This gives permission to execute an activity by an agent in authority.

Communication-with-Commitment links are described by means of:

- *Committing-Agent:* The agent that has authority.
- *Committed-Agent:* The agent that acquires obligations during communication.
- *Committing-Agent-Role:* the role of the Committing-Agent for which authority is defined
- *Committed-Agent-Role:* the role of the Committed-Agent for which authority is enabled.
- *Authority-for:* the collection of speech acts defining the authority of the Committing-Agent. These speech acts can mention contents or content templates that define the classes of messages acceptable for the communication with authority.

Winograd and Flores [24] describe the language/action approach in which commitment creating communication plays the central role.

4.0 Credibility (competence)

The conflict management model we discuss in the next section is based on an organizational definition of agent credibility or competence.

The question here is: given two conflicting beliefs of two agents, which of them is more "believable"? We answer this question by defining a partial order of beliefs ($<_c$) held by different agents. This order is defined on the basis of the roles agents act in. The definition of credibility is the following:

If

(i) b1 is a belief assumed by a1 while acting to achieve the goal of role r1, such that b1 is required to achieve this goal, and

(ii) b2 is a belief assumed by a2 while acting to accomplish the goal of role r2, such that b2 is needed to achieve this goal, and

(iii) b1 is in conflict with b2,

then

b1 is more credible than b2 iff according to the partial order of credibility (a2 r2)$<_c$ (a1 r1).

This notion of credibility allows us to state things like:

- Agent John is more credible (competent) as a senior C programmer than agent Tom as a COBOL programmer (different agents, different roles). This is written as (Tom COBOL-Programmer)$<_c$(John C-programmer). The implication is that John's beliefs held as a C-Programmer are more credible as Tom's beliefs held as a COBOL-Programmer.
- Agent John is more credible (competent) as a programmer than agent Tom (different agents, same role). This is written as (Tom Programmer)$<_c$(John Programmer) and implies that John's beliefs as a Programmer are more credible than Tom's beliefs as a Programmer.
- Agent John is more credible (competent) as a programmer than as a system administrator (same agent, different roles). This is written as (John System-Administrator)$<_c$(John Programmer) and implies that John's beliefs as a Programmer are more credible than his beliefs as a System-Administrator.

4.1 Ideal credibility

A notion of *ideal* credibility can be defined by considering the credibility relations among agents in roles that imply responsibility for goals and their subgoals.

The partial order defining authority of belief is ideal iff

agent(a1) & has-role(a1, r1) & has-goal(r1, g1) &

agent(a2) & has-role(a2, r2) & has-goal(r2, g2) &

has-subgoal(g1, g2) =>

(a1, r1) > (a2, r2).

In words, if agents a1 and a2 have roles r1 and r2 with goals g1 and g2 respectively, and g2 is a subgoal of g1, then a1 in role r1 must be more credible than a2 in role r2. In organizational terms, agents responsible for higher order goals should be more credible than the agents responsible for the subgoals of these goals. This may be used for example to determine who should fill managerial or team leading positions.

It is for further study to decide how important are this ideal conditions in real organizations and to what extent they can be enforced. In this paper we are only interested in credibility as a basis for the conflict management model described in the next section.

4.2 The relation between authority and credibility

Ideally, authority (of action) should not contradict credibility. If agent a1 has authority of action over agent a2, an action of a1 should not force a2 to contradict one of its beliefs B, unless a1 is more competent than a2 as well. In organizational terms, agents that manage other agents should not force them to act against their beliefs unless the manager agent is more competent (credible) in the respective issue.

4.3 Using organizational models

Agents use this ontology to build models of the other agents they cooperate with in an organization as well as to represent themselves. The self representation of an agent contains its roles, goals, activities and communication links. This representation is assumed to be complete and accurate. The representation of other agents consists of the same elements, but is *not* assumed to be either complete or accurate. During problem-solving and cooperation, agents may update their models of other agents. In particular, agents detect conflicts among their goals and other agents' goals and this triggers negotiation for avoiding conflicts. Agents plan their cooperative actions depending on their model of the agents they interact with.

5.0 Conflict management with the credibility/deniability model

In a multi-agent reasoning system encountering contradictions like *p&q=>false* is a natural thing to happen. Often, the agent that encounters the contradiction has to eliminate it by retracting some current belief that supports either *p* or *q*. This section addresses the issue of how to determine which of the possible supporting beliefs to retract.

5.1 Types of conflict

In our framework we distinguish among several kinds of conflicts:

- *terminological conflicts* arise from inconsistent terminologies. For example, defining an engine as `(and v6engine l4engine)`.

- *assertional conflicts* arise from the inconsistent use of terminology in models. For example, asserting `(v6engine e)` and `(l4engine e)`.
- *temporal conflicts* are a subclass of assertional conflicts that arise from inconsistent use of time in models. For example, asserting that `(cause ev1 c1)[sept 94]` and `(effect ev1 f1)[august 94]`.

Because our representational substrate is able to verify the consistency of the employed terminologies, we have focused on a conflict management model that can deal with the latter two types of conflicts.

5.2 Beliefs, premises, producers and consumers

Agents represent their beliefs as *propositions*. Agents communicate by sending and receiving messages whose content layer consists of propositions. Propositions are formed with concepts from a *common ontology* shared by all agents. This ontology is a a *concept* and *role* taxonomy encoded in the description logic language. If `e1` is an automobile engine used by the enterprise, then the proposition `(v6engine e1)[13 march 94]` expresses the fact that `e1` is believed to be a v6 engine at 13 march 94, while `(l4engine e1)[14 march 94]` expresses the fact that `e1` is believed to be an l4 engine at 14 march 94. Properties of objects are described by propositions such as `(power e1 132)[13 march 94]` - the power of engine e1 is believed to be 132 at the given time - or `(torque e1 150[13 march 94]` - the torque of e1 is believed to be 150 at the specified time. In these examples, `v6engine` and `l4engine` are concepts, while `power` and `torque` are roles. Any agent maintains two kinds of propositions. *Premises* are propositions sent to the agent by other agents that consider them true. The agent has no access to whatever justification the sending agent may have for the proposition. *Derived propositions* (or simply propositions) are propositions inferred by the agent based on the available premises and on the agent's knowledge of the domain. For example, being told that `(v6engine e1)[13 march 94]`, the agent may derive `(heavy e1)[13 march 94]` based on domain knowledge. Agents that supplied propositions are named *producers* of that information. Agents that have received propositions are named *consumers* of the information.

5.3 Credibility and deniability

When information is integrated from multiple sources, contradictions can easily occur. For example, the Marketing Agent may have determined that for a new automotive product a v6 engine would sell better. Hence marketing will sent the Management Agent a message telling that the engine should be a v6: `(v6engine e1)[starting 13 march 94]`. From different requirements, the Design Agent may determine that only a l4 engine can be used: `(l4engine e1[starting 14 march 94]`. Using domain knowledge that the `v6engine` and `l4engine` concepts are disjoint, the Management Agent will derive a contradiction (for the common time interval during which both beliefs are held):

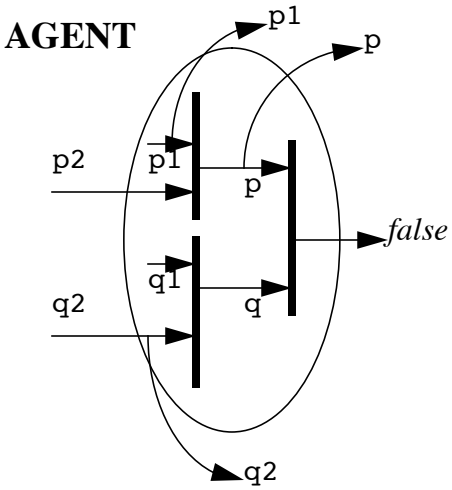
```
(and
  (v6engine e1)[starting 13 march 94]
  (l4engine e1)[starting 14 march 94]
  => false.
```

The conflict management service tries to remove this contradiction by considering two properties of information, *credibility* and *deniability.*

Credibility (or *competence*) is defined like in section 4, based on a partial order of agents in given roles. Any belief originating from an agent a in role r1 is more credible than a belief originating from agent b in role r2 iff (b r2) $<_c$ (a r1), where "$<_c$" is the partial order. We extend the use of this notation on beliefs, writing b1 $<_c$ b2 when b1 and b2 are beliefs originating from agents among which $<_c$ is defined. Because the order is partial, not all beliefs are comparable.

Credibility is an order relation (reflexive, transitive and anti-symmetric). We can assign to each belief b a numerical value n(b) such that for any two comparable beliefs b1 and b2, b1 $<_c$ b2 $\leftrightarrow$ n(b1) $\leq$n(b2). This can be done simply in the following manner. First, all beliefs that are the least credible (no other belief is less credible) are assigned 0. Then, all beliefs that are directly preferred to these (i.e. those b such that for an a in the previous set a $<_c$ b) are assigned 1. This is repeated for all beliefs (assumed to be in finite number). In the end, if a belief is assigned more than one value, the largest value is kept. This process makes it possible to derive an order preserving numerical rating for credibility.

After information is delivered to the interested agents, these consumers will use it to make decisions and take action. For example, if marketing was first to determine that the engine must be a v6, it might have sent this information to the Purchasing Agent. The Purchasing Agent used the information to order v6 engines from another company. Later, design discovered that the engine must be a l4. If the design view is accepted, purchasing will have troubles in cancelling the order (paying penalties, etc.). This shows that information that has been consumed may be costly to retract later. We define the *undeniability* of consumed information as a measure of the cost to retract it (high undeniability means high costs). We often use *deniability* as the inverse of undeniability. Undeniability (or deniability) is determined by the consumers of information. We make the assumption that agents are honest when assessing undeniability and do not use this to minimize their own workload.



p1, q1 – internally created by agent

p2, q2 – imported (with credibility)

p,q – derived by agent

p1, p, q2 – supplied to other agents for consumption

Undeniability(p1)=ConsumerCost(p1) + ConsumerCost(p)

Undeniability(q2)=ConsumerCost(q2)

p&q=> false (contradiction)

{p1, p2, q1, q2} = conflict set

FIGURE 6. Types of beliefs in conflict management.

The various sorts of beliefs we distinguish among are illustrated in figure 6.

In conclusion, when a contradiction *p&q=> false* is encountered, we need to retract either *p* or *q* in order to remove the contradiction. The

decision of what to retract considers the credibility of the producers of the *premises* from which *p* and *q* were inferred as well as the cost incurred upon the agents that have consumed any propositions derived from the premise to be retracted. To ensure accuracy, we assume that the partial credibility relation is accurately defined and that consumers honestly assess deniability costs.

5.4 The c-u space

The conflict resolution process has the goal of retracting one or several *premises* so that the contradiction can not be derived any more. When a premise is considered for retraction, both the credibility of the agent producing it and the deniability of the propositions that are supported by the considered premise (and hence will be retracted with the considered premise) are considered.

Suppose we have determined a set $\{p_i\}$ of premises which supports a *p&q=>false* contradiction. To each p_i we can attach:

- a *credibility measure*, the numeric credibility previously introduced and
- an *undeniability measure* - derived from the sum of deniability costs of all propositions that would have to be retracted if p_i is retracted.

A high credibility means that the proposition is more difficult to retract since a higher competence has to be contradicted (or otherwise put there exist less credible candidates). A high undeniability means that the proposition is more difficult to retract because the costs of retraction incurred upon consumer agents will be great.

We can represent these two values in a diagram having credibility on the *x*-axis and undeniability on the *y*-axis. Such a diagram is called a *c-u space* and is illustrated in figure 7.

Propositions from the c-u space that have both low credibility (e.g. 0) and low undeniability are easy to retract because they are less credible and do not incur significant costs. Propositions that have high credibility and high undeniability are hard to retract exactly for the opposite reasons. An aggregated measure of both credibility *and* undeniability is the distance *r* to the origin. If a proposition with high credibility or undeniability is considered for retraction, retraction must be negotiated with the producer and/or consumers. If credibility is high, the producer must agree with the retraction. If undeniability is high, consumers must be pooled and a measure of their approval must be computed.

We can represent regions in the a-u space for the classes of propositions that can be retracted with or without negotiation. To do this, we introduce a threshold value of credibility, c_t, and a threshold value of undeniability, u_t, such that propositions having credibility and respectively undeniability higher than the threshold value can be retracted only after negotiation. Figure 7 shows the regions defined by these thresholds in the c-u space. [23] and [25] are examples of work exploring negotiation as a means to mediate among conflicting agents.

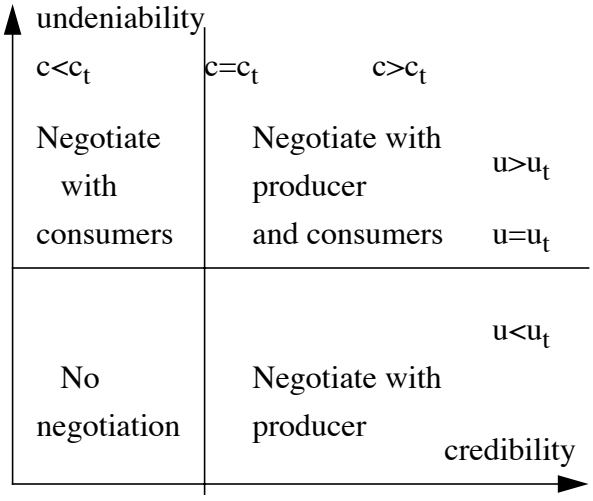


FIGURE 7. Negotiation regions in the c-u space.

5.5 The model

Putting together the above discussed elements we can formulate the following model for contradiction removal:

Given: a contradiction of the form: *p&q=>false*

1. Determine the support set of *p*, that is the set of premises *p* is derived from, and the support set of *q*, that is the set of premises *q* is derived from. Together, these two sets form the conflict set.

2. Group the propositions from the conflict set into 4 sets corresponding to the 4 negotiation regions. In each region, order the component propositions in increasing order of the value of $r = (a^2+u^2)^{1/2}$

3. Considering the 4 regions in the order: (1) no-negotiation, (2) negotiation-with-producer, (3) negotiation-with-consumers, (4) negotiation-with-both, take each proposition in order and try to retract it:

   - If the premise falls in the no-negotiation region, retract it
   - If the premise falls into a negotiation region, negotiate with the required agents.

4. Retract the first proposition that passes the above tests and check if after retraction the contradiction can be rederived. If so, repeat the procedure. If no proposition can be retracted, report failure.

Initially, a conflict is discovered by an agent that derives a contradiction. The conflict management model then retracts one (or more) premises that led to the contradiction. Since the retracted premises can be themselves beliefs held by some other agents, their retraction may trigger other retractions in the agents holding them. These propagated retractions are performed using the same model - hence we have a propagated chain of retractions. Further work is needed to understand what to do if these chains are circular or too long.

5.6 Negotiation with or without an arbiter

One simple solution for negotiation would be to take each belief from the conflict set and see if the interested parties agree to give it up. In this case the reached decision (if any) is made only on the basis of the local perspectives of agents. A good decision may require however a more global or strategic perspective over the problem, e.g. the perspective that a top manager would bring into the discussion.

The organization model helps us find an agent that has this perspective. Since each belief from the conflict set is tagged with the (agent role) specification of its producer, an appropriate arbi-

ter can be defined as one that is responsible for an organization goal that subsumes most (or all) goals of the roles of the agents that originated the conflict set.

Competent arbiter agents retrieved in this way can have negotiation knowledge, expressed as conversations, that can be used to handle conflicts. In this approach, the agent discovering the conflict finds an arbiter and presents it the c-u annotated conflict set. The arbiter takes over negotiation and finds a resolution.

5.7 Further issues

A number of more detailed issues have to be dealt with to make the model fully operational:

1. *Retracting from negotiation-with-producer region first*. Because the credibility measure is organization-wide defined and changes slower in time, it may be more accurate than the deniability measures. Due to this feature, we choose to attempt to retract the propositions from the negotiate-with-producer region before those from the negotiate-with-consumers region.

2. *When is undeniability specified?* This raises some problems. The "most appropriate" time for specifying undeniability is when the proposition has to be retracted. This would imply more communication overhead at conflict management time to determine undeniability of each belief. We have chosen a solution that allows retraction without this overhead by allowing consumers to send producers incremental updates of retraction costs as they become known. In this way costs will be accurate when conflict management is invoked.

3. *How is undeniability specified?* In implementations we have versions using either boolean measures (*false* = "can deny", *true* = "better don't deny") or numerical measures (like the 1 .. 10 range).

4. *How are undeniability costs aggregated?* The measure of a proposition's undeniability must be aggregated from the deniability costs provided by each consumer. If a proposition has *n* consumers (directly or indirectly, through propositions derived from it), then a normalized cost can be obtained by averaging the costs reported by the consumers.

5. *Equivalence classes*. Propositions with the same values for authority and undeniability form equivalence classes. For propositions in negotiable regions the implemented algorithm tries *all* propositions in an equivalence class in order to retrieve the "most retractable" proposition (the one reported as most acceptable for retraction by its producer and consumers). For propositions in the non-negotiable region, a random choice is made.

6. *Handling time mapped propositions*. Our representation language allows time mapped propositions. These propositions mention a time interval during which they are true or false. In this case, a contradiction $p\&q \Rightarrow false$ exists iff p and q are true on a common subinterval. When retracting a premise, it is enough to guarantee that either p or q will be retracted on the common subinterval causing the contradiction. The TMS we use is extended to handle time-mapped propositions and can retract propositions on subintervals.

7. *Early detection of contradictions*. It is important to detect contradictions as early as possible. With a reasoning system that reasons backwards (inferring a proposition only when asked to do so) contradictions amongst propositions that are never queried may

remain undetected. Because of this, we are using a truth-maintenance system that works forward, inferring all possible propositions whenever new premises are added.

6.0 Final remarks

The model of belief revision presented here has three main advantages. First, it is accurate because the selection of the retracted belief is based on the views of all involved parties at the moment the contradiction is detected. Second, this means that the selection implicitly relies on domain knowledge and on the current state of the global problem-solving effort. Third, by estimating costs and identifying negotiation regions, the model takes advantage from situations in which negotiation may not be required or in which a smaller amount of negotiation may suffice.

As similar research is concerned[1], Petrie [19] was the first to introduce reasoning about retraction in a TMS, in a manner that introduced domain dependent criteria. His work was carried out in the context of Doyle's JTMS [9] that allows non-monotonic justifications. Our work uses a McAllester TMS [2] that forbids non-monotonic justifications and has a more efficient (linear) labeling algorithm. Both of these use single context TMSs. It would be interesting to see how the authority/deniability model can be integrated in a multiple-context TMS as the ATMS [8]. Since the ATMS stores with each datum its complete set of prime implicants, the reasoner can simultaneously consider all candidate premises for retraction. In the single context case we can only use the current justifications and whatever premises support them (although other premises and justifications may exist as well). Because of this, in the multiple-context case we may be able to avoid step 4 of our retraction algorithm (trying to rederive the contradiction and repeating the procedure for any new justifications of the contradiction) by retracting everything that has to be retracted from the beginning.

1. Other efforts have tried to maintain consistency in distributed TMS environments by devising distributed labeling algorithms. Examples are [7] for the JTMS and [16] for the ATMS [8]. Our problem however is not distributing the TMS algorithm as we make no assumption about the problem-solving mechanisms of the agents.

7.0 Acknowledgements

This research is supported, in part, by the Manufacturing Research Corporation of Ontario, Natural Science and Engineering Research Council, Digital Equipment Corp., Micro Electronics and Computer Research Corp., Spar Aerospace, Carnegie Group and Quintus Corp.

8.0 References

1. Allen, J., F., (1983) Maintaining Knowledge About Temporal Intervals, Communications of the ACM, 26 (11), pp 832-843, 1983.

2. McAllester, D., (1990) Truth Maintenance, Proceedings AAAI-90, pp. 1109-1116.

3. Barbuceanu, M. (1993) Models: Toward Integrated Knowledge Modeling Environments, Knowledge Acquisition 5, pp. 245-304.

4. Barbuceanu, M., Fox, M.S., (1994) The Information Agent: An Infrastructure Agent Support-

ing Collaborative Enterprise Architectures, to appear in Proceedings of the 3-rd Workshop on Enabling Technologies: Infrastructures for Collaborative Enterprises, IEEE Computer Society Press.

5. Borgida, A., Brachman, R. J., McGuiness, D. L., Resnick. L. A., (1989) CLASSIC: A Structural Data Model for Objects, Proc. 1989 ACM SIGMOD International Conference on Management of Data, june 1988, pp. 59-67.

6. Brachman, R. J., Schmolze, J. G., (1985), An Overview of the KL-ONE Knowledge Representation System, Cognitive Science 9(2), April-June 1985, pp. 171-216.

7. Bridgeland, D. M., Huhns, M., N. (1990) Distributed truth maintenance, Procedings of AAAI-90, pp 72-77

8. DeKleer, J. (1986) An Assumption Based TMS, Artificial Intelligence, 28(2), pp 127-244, march 1986

9. Doyle, J., (1979) A truth maintenance system, Artificial Intelligence, vol 12, no. 3, pp 231-272

10. Finin, T., et al. (1992) Specification of the KQML Agent Communication Language, The DARPA Knowledge Sharing Initiative, External Interfaces Working Group.

11. Fox, M. S., (1993) A Common-Sense Model of the Enterprise, Proc. of Industrial Engineering Research Conference, 1993.

12. Galbraith, J., (1977) Organization Design, Addison Wesley Publishing Company.

13. Genesereth, M. R., Fikes, R.E. (1992) Knowledge Interchange Format, Version 3.0, Reference Manual, Computer Science Dept., Stanford University, Technical Report Logic-92-1.

14. MacGregor, R., Bates, R., (1987) The LOOM Knowledge Representation Language, ISI IRS-87-188, USC/ISI Marina del Rey, CA

15. Luck, K., Nebel, B., Peltason, C., Schmeidel, A., (1987) The Anatomy of the BACK System, KIT Report 41, Fachbereich Informatik, Technische Universitat Berlin, Berlin.

16. Mason, C., Johnson, R.,R., (1989) DATMS: a framework for distributed assumption based reasoning. In Les Gasser and Michael N. Huhns, editors, Distributed Artificiall Intelligence, Volume II, pp. 293-317, Pitman Publishing, London, 1989

17. Mintzberg, H., (1983) Structure in Fives: Designing Effective Organizations, Prentice Hall International Editions.

18. Pan, J.Y.C., Tenenbaum, J. M., (1991) An Intelligent Agent Framework for Enterprise Integration, IEEE Transactions on Systems, Man and Cybernetics, 21, 6, pp. 1391-1408, 1991.

19. Petrie, C., (1987) Revised dependency-directed backtracking for default reasoning, Proceedings of AAAI-87, pp 167-172

20. Roboam, M., Fox, M. S., (1992) Enterprise Management Network Architecture, A Tool for Manufacturing Enterprise Integration, Artificial Intelligence Applications in Manufacturing, AAAI Press/MIT Press, 1992.

21. Schmolze, J.G., (1985) The Language and Semantics of NIKL, BBN Inc., Cambridge, MA.

22. Shoham, Y., (1993) Agent-Oriented Programming, Artificial Intelligence 60 (1993) pp 51-92.

23. Sycara, K., (1989) Multi-agent compromise via negotiation, In Les Gasser and Michael N. Huhns, editors, Distributed Artificiall Intelligence, Volume II, pp. 119-137, Pitman Publishing, London, 1989

24. Winograd, T. and Flores, F., (1986) understanding Computers and Cognition: A New Foundation for Design, Ablex Publishers, 1986.

25. Zlotkin, G., Rosenschein, J.S. (1989) Negotiation and task sharing among autonomous agents in cooperative domains, Proceedings of IJACI-89, pp. 912-917, Detroit, MI, aug. 1989