Mediated Conflict Recovery by Constraint Relaxation

J. Christopher Beck[1]
chris@cs.utoronto.ca
(416) 978-7321

Mark S. Fox
msf@ie.utoronto.ca
(416) 978-6823

Department of Computer Science, University of Toronto
10 King's College Rd., Toronto, CANADA, M5S 1A4
fax: (416) 978-3453

Abstract

Dynamic events in a multiagent environment can be a source of conflict. We model the network of inter-agent commitments as a constraint graph. Conflicts arise when an event prevents fulfillment of a commitment, resulting in an infeasible constraint graph. Constraint relaxation directed by a mediating agent is used to reconfigure the commitment graph. We investigate this general approach in the domain of supply chain management and present a schema for constraint relaxation algorithms. Experimental results on Partial Constraint Satisfaction Problems (PCSPs) and schedule optimization are given along with a sketch of the conflict recovery protocol in development.

Keywords: plan execution, conflict detection and recovery, mediation, constraint relaxation

1.0 Supply Chain Management

The ability to quickly respond to environmental changes is recognized as a key element in the success and survival of corporations in today's market [Nagel 91] . This agility includes an ongoing monitoring of events both inside and outside the corporation, quick recognition of the impact of exogenous events, and rapid re-planning and reconfiguration to allow the enterprise to take advantage of opportunities and minimize costs.

In a manufacturing enterprise, the entire supply chain is subject to unexpected, conflict-causing events. Exogenous events are many and varied: change in the customer order, late delivery or price change of a particular resource, machine breakdown, an urgent order from a good customer, and so on. Handling these events requires cooperation among sales, marketing, accounting, material planning, production planning, production control, and transportation.

The departments and factories in a traditional manufacturing enterprise are encapsulated into software agents. We represent the interactions of the agents in the supply chain as a commitment/constraint graph. In the high-level planning or scheduling in the supply chain, each agent is assigned one or more tasks toward the global goal. The tasks are subject to constraints such as precedence constraints (e.g. one task must be performed before another on is started) and resource constraints (e.g. a task must use a particular raw material or machine-type). Because of the distribution of knowledge, abilities, and resources in the supply chain, a schedule for any non-trivial group of tasks will assign inter-dependent activities to disparate agents. In order to correctly execute the tasks, agents must commit to satisfying the constraints on the tasks. Therefore, a constraint graph is created *among the agents* based on the distribution of tasks.

Figure 1 shows an example of the creation of a commitment graph in a manufacturing enterprise. The original process plan (Figure 1A) represents a set of inter-related tasks. When a task is assigned to an agent (Figure 1B), the agent commits to the satisfaction of the constraints on the task. The commitment graph is a distributed constraint graph: the satisfaction of a task's constraints is the only way an agent can fulfill its commitments. Given multiple orders of inter-related activities, a full constraint graph will grow to a non-trivial size. A search for a near-optimal reconfiguration has to handle the combinatorial explosion of interdependent alternatives in resource choice, transportation method, and execution times.

1. Supported by Natural Sciences and Engineering Research Council Centennial Fellowship.

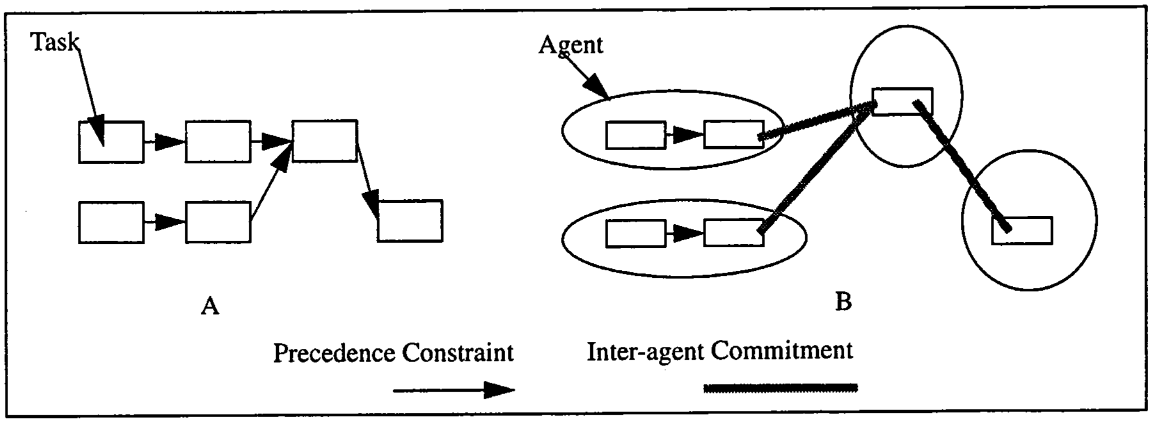


Figure 1. Constraints on Tasks Become Commitments Among Agents

Finding an original, feasible graph is the problem of multiagent planning. We assume that such a plan (and the corresponding graph) is already in place. Our focus is the recovery from stochastic events that make the agent interactions infeasible. A conflict, then, is represented by a constraint graph where one or more constraints can not be satisfied. Recovery necessitates assessment of alternate graph configurations and adoption of the one with the least negative global impact. Figure 2 presents the commitment graph shown above after an event has occurred at one of the agents. The event prevents a task from being completed on time, resulting in the failure by the agent to fulfill a commitment. Notice that other indirectly dependent commitments also cannot be met after the unexpected event.

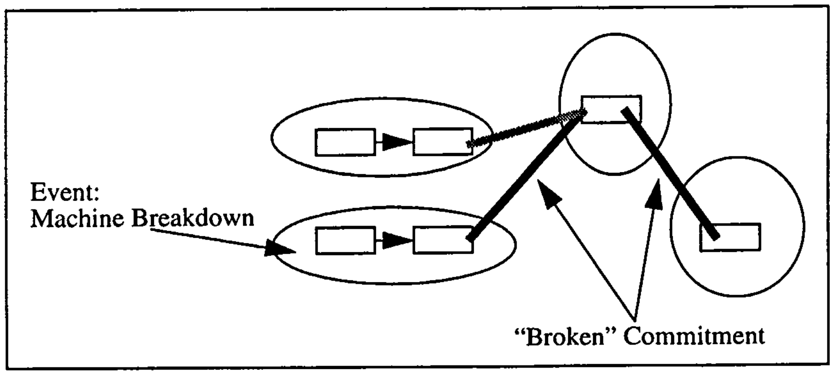


Figure 2. An Unexpected Event Causes a Cascade of Unfulfilled Commitments

The conflict recovery process is hierarchical. For example, in modeling multiple production centers (e.g. factories), conflicts among agents within one center are at a level of abstraction below the conflicts among the centers. It is possible, however, for an event within a production center to have non-local effects. If the production center can not reconfigure its local graph in order to continue to meet external commitments, the event leads to conflict among production centers.

The Enterprise Integration Laboratory at the University of Toronto is pursuing the development of an Integrated Supply Chain Management (ISCM) system addressing these and other problems. The project is based on a distributed simulation of an enterprise. The ISCM simulation is operational and the functionality of the individual agents is being developed to enable exploration of the conflict management issues. The balance of this paper presents a schema for generalized constraint relaxation and results of centralized relaxation algorithms on two sets of problems from the literature. We conclude with a discussion of our future work using a mediator to apply constraint relaxation algorithms to conflict management in the supply chain.

2.0 Constraint Relaxation

2.1 Introduction

Given our representation of a conflict as an infeasible commitment graph, conflict recovery is a reconfiguration of the graph to re-establish feasibility. This reconfiguration is constraint relaxation as a subset of commitments may have to be modified (at some cost) in order to find a conflict-free graph.

A constraint relaxation algorithm takes an overconstrained constraint graph and attempts to find the minimum cost modification to be made to a subset of constraints to produce a feasible graph. The cost associated with modification of the constraints is the optimization criteria. Relaxation is not simply choosing to ignore a constraint. *A constraint can be relaxed in many ways and how the constraint is relaxed has significant impact on the cost and utility of the modification.* If we relax the constraint on the due date of an activity to allow three more hours, the incurred cost will be significantly less than if the relaxation allows three more days.

We have created a schema that defines a family of incomplete relaxation algorithms based on propagation of information through the constraint graph. By varying the heuristics used in the instantiations, algorithms can be created for specific problem types. For example, on small problems the relaxation algorithms might search through the entire constraint graph. On larger problems, where time and space complexity are an issue, we can limit the propagation and execute the relaxation algorithm multiple times on different subgraphs. Obviously, limiting the search also impacts the performance of the algorithm.

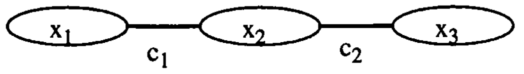
We extend the common constraint model used in CSPs by adding two functions to the constraint representation:

- **GenerateRelaxation**, which returns a set of constraints that are relaxations of the constraint.
- **RelaxationCost**(c_k), indicating the local cost of relaxing a constraint to c_k.

2.2 Example

The central mechanism for investigation of relaxation costs is the propagation of information over constraints. Propagation is common in consistency algorithms, however the constraint graph structure allows information other than simply the values to be "transmitted" to other variables.[2] Here we will give a brief example of the propagation of values, costs, and relaxations with the constraint graph in Figure 3.

Variable Domains: $\{D_1, D_2, D_3\} = \{\{1,2\},\{1,2\},\{1,2\}\}$



c_1: Domain = $\{(1,1),(1,2)\}$		c_2: Domain = $\{(2,1),(2,2)\}$	
Relaxed Domain	Cost	Relaxed Domain	Cost
rc_1: $\{(1,1),(1,2),(2,1)\}$	1	rc_3: $\{(2,1),(2,2),(1,2)\}$	1
rc_2: $\{(1,1),(1,2),(2,2)\}$	3	rc_4: $\{(2,1),(2,2),(1,1)\}$	2

Figure 3. A Simple Constraint Graph

Suppose we want to find the minimum cost if $x_1 = 2$. In propagating the value $x_1 = 2$ to x_2, c_1 can not be satisfied as is. The **GenerateRelaxation** function, in this case, simply returns a singleton set containing the minimum local cost relaxation (rc_1). Relaxation rc_1 is chosen and the value 1 is propagated to x_2. Variable x_2 is assigned 1 and the same greedy relaxation procedure is performed on c_2. Relaxation rc_3 is chosen and the value of 2 is propagated to x_3. Once at x_3, the cost of the relaxations is propagated backward. A cost of 1 is propagated from c_2 to x_2. This is summed with the cost at c_1 and propagated to x_1. The cost of the graph, with these relaxations is 2. If the cost is acceptable we propagate the relaxation in the same way as we propagated the values. With relaxation propagation we actually replace each constraint with the best relaxation that was tried in the cost/value phase.

2. This use of propagation builds on propagation of preferences [Sadeh 89]. In that work, values are propagated though the constraint graph and the preferences that each variable has for the propagated value is propagated back. A more global view of the utility of local values is formed at the starting variable. We propagate costs rather than preferences and the relaxation of constraints as the source of the cost.

2.3 The Relaxation Schema

Our constraint relaxation schema identifies four heuristic decisions points:

1. Selection of a *source variable*. The source variable is the origin of value propagation.
2. Selection a set of *candidate values* at each variable visited by value propagation. The candidate values are the elements of the domain of the variable that are searched over to find a close-to-optimal graph reconfiguration.
3. Selection of a set of *outgoing constraints* at each variable. The outgoing constraints define the constraints along which value propagation will proceed. A key method of dealing with the complexity of the search is the limitation of the propagation to a small subgraph.
4. Selection of a set of *candidate constraints* at each outgoing constraint. The set of candidate constraints are those relaxations (plus the constraint itself) that we will search over in attempting to find a low cost feasible graph.

These decision points create an exponential backtracking-like search. Practical algorithms exploit the problem structure (assessed via texture measurements[3]) to limit the size of the sets at each of the three latter decision points. If there is a possibility of cycles in the graph, they must be dealt with to ensure termination.[4]

The pseudocode below defines a search through the constraint graph where the global impact of a number of values and relaxations are tested. The algorithm (aside from the selection of a source variable) is repeated at every variable reached in the value propagation. Again note that propagation is the mechanism for the transmission of information upon which search decisions are made.

Pseudocode for our constraint relaxation schema follows:

```
select a variable and a set of candidate values
for each value
   select a set of outgoing constraints
   for each outgoing constraint, c
      propagate the value to the constraint
      select a set of candidate constraints, $CC_c$
      for each candidate constraint
         propagate value along the constraint and record the cost returned
      record the element of $CC_c$ that returned the minimum cost
   store the sum of the minimum costs from each outgoing constraint
if one of the costs is acceptable
   select corresponding local value
   instantiate the value
   propagate relaxations along the outgoing constraints
```

2.4 Relaxation on Small Problems

We have applied algorithms within this schema to Partial Constraint Satisfaction Problems (PCSPs) [Freuder 92]. The algorithms are exponential in complexity and require significant caching of information in order to increase performance and cope with graph cycles. Despite this, the algorithms perform well compared with PEFC3, the best PCSP algorithm investigated by [Freuder 92].

Each problem set contains ten problems. Problem Set 1, 3 and 5 are respectively 10-, 12- and 16-variable PCSP problems created by [Freuder 92]. The cost of an unsatisfied constraint is 1. Problem Sets 2, 4, and 6 respectively contain the same problems as Sets 1, 3, and 5 except the cost of not satisfying a constraint is assigned randomly for each constraint on the [1,9] interval. The relaxation algorithms are instantiations of the propagation-based relaxation schema described above, with varying parameters.[5]

3. Texture measurements [Fox 89] [Sycara 91] are techniques that assess structural properties of the constraint graph representation of the problem. Based on these measurements, heuristic search-guiding decisions can be made.

4. A cycle detection mechanism is presented in [Beck 94].

5. The parameters specify the actions at the decision points. For further information see [Beck 94].

The number of consistency checks is used as a measure for the work done by each algorithm. A consistency check accesses the constraint definition to determine if the constraint is satisfied by the current variable assignment. In most cases, the relaxation algorithms ran significantly faster than PEFC3 (Figure 4, note the log scale on the vertical axis). No results could be found for the MMV algorithms on Sets 5 and 6 due to exponential memory requirements.

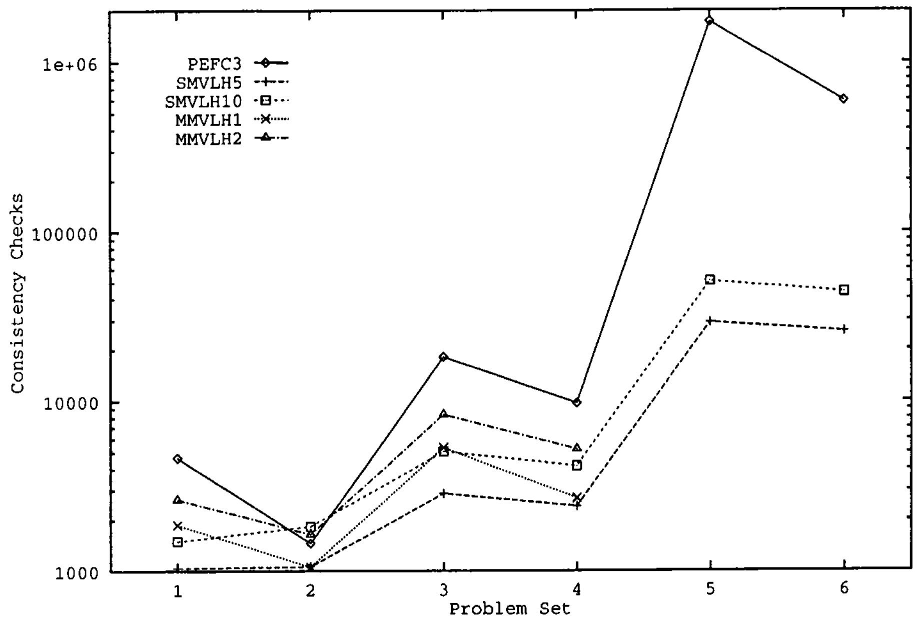


Figure 4. Number of Consistency Checks for each Algorithm

The PEFC3 algorithm is complete and therefore is guaranteed to find the minimum cost solution. Because the relaxation algorithms are incomplete we also compared the cost of the solutions found by each algorithm (Figure 5). It can be seen, that the relaxation algorithms find a close to minimum cost across the problem sets.

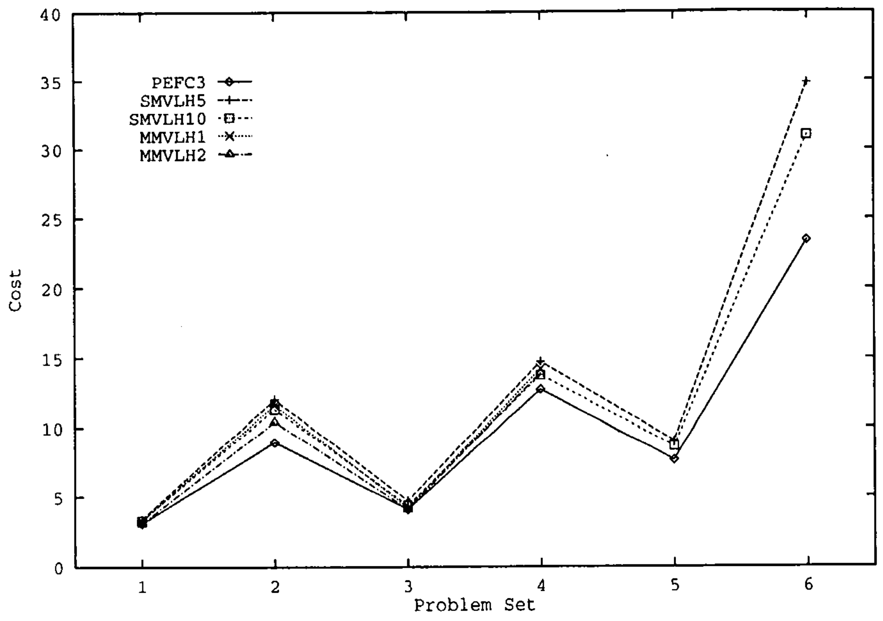


Figure 5. Costs of the Solutions Found by each Algorithm

The weakness of the both the relaxation algorithms tested above and the PCSP algorithms is their exponential complexity. None of these algorithms will scale-up to problems much larger than 15 variables.

2.5 Relaxation on Larger Problems

In order to apply relaxation algorithms to problems of practical size, we modulate the search by manipulation of the decisions made at the heuristic decision points. The intuition behind this approach is to limit the both the size of the graph on which relaxation is executed and to limit the number of alternatives that are searched over. These heuristic decisions are achieved by limiting the size of the sets of outgoing constraints, candidate values, and candidate constraints. The basis for these heuristic decisions are texture measurements revealing properties of the constraint graph.

We have created a schedule optimization algorithm that uses three instantiations of the relaxation schema and a number of texture measurements. Space limitations preclude a full description, however the main loop of the algorithm is composed of the repeated execution of relaxation algorithms on different small subgraphs identified using texture measurements. Relaxing small subgraphs avoids the difficulties of the exponential complexity on large problems.

For our experiments, we use a 8 sets of schedule optimization problems from the literature [Sadeh 91] . Each set is composed of 10 problems of 20 orders, 5 resources, and 5 activities per order. The constraint representation of these problems forms a graph of over 150 variables. The optimization criteria is based on both tardiness and inventory costs.

We compare the relaxation algorithm to the MICRO-BOSS optimizing scheduler and a dispatching scheduler. MICRO-BOSS is a constructive scheduler using a micro-opportunistic approach to identify important trade-offs in the scheduling problem [Sadeh 91] . MICRO-BOSS implicitly represents the relaxation of due date constraints by attaching a cost factor to the tardiness of each order. Tardiness is part of the explicit overall cost representation which also includes inventory costs. The dispatching scheduler is a repair-based algorithm where the main decision criteria is the duration of conflicting activities [Beck 94] .

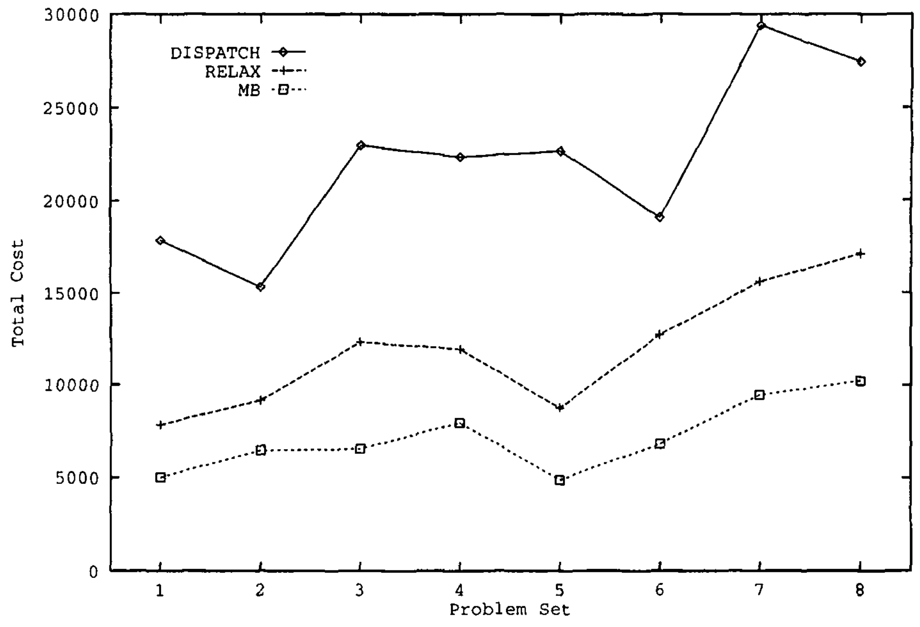


Figure 6. Average Schedule Cost Found by each Schedule Optimization Algorithm.

Figure 6 indicates that MICRO-BOSS performed better that the relaxation algorithm across all problem sets.[6] While the relaxation-based algorithm outperforms a dispatching rule, it is clearly inferior to MICRO-BOSS. We believe that the main reason for this is precisely the strength of MICRO-BOSS. The micro-opportunistic approach uses texture

6. The results presented for MICRO-BOSS are representative of one system configuration. MICRO-BOSS is an evolving system and these results should be viewed as a snapshot [Sadeh 94] .

measure to re-evaluate trade-offs at each step in the scheduling process. The relaxation algorithm uses less sophisticated texture measures that in some cases focuses on subgraphs rather than on single tasks. The knowledge of the costs of trade-offs in MICRO-BOSS is analogous to knowledge of the cost of competing relaxations in the relaxation algorithm. The MICRO-BOSS algorithm integrates this knowledge more deeply into its scheduling decisions leading to the increased performance seen in the experiment.

2.6 Using Relaxation For Conflict Recovery

The generality and flexibility of the relaxation schema is demonstrated by these two experiments. Algorithms within the schema perform well on PCSPs which are based on a general constraint representation. In more specific domains, relaxation algorithms are also applicable however do not perform as well as algorithms explicitly designed for the narrower domain. The fact that algorithms within the schema can be used in schedule optimization as well as in the PCSPs shows that the schema is general to the constraint representation and not dependent upon any particular domain or constraint type.

The isolation of the heuristic decision points allows a wide flexibility in the type of algorithms that can be instantiated. The poor performance in the scheduling domain can be traced to the specificity of the MICRO-BOSS approach and the relatively general techniques used in the relaxation algorithm. We conjecture, that more specific instantiations of the schedule algorithm, perhaps incorporating a relaxation-based interpretation of the MICRO-BOSS algorithm, will reduce the performance gap.

We believe that the generality and flexibility of the schema make it an excellent tool for conflict management in a multiagent environment. Not only can all types of constraints be accommodated by an algorithm, but also the algorithm can be tuned to differentially focus on particular types of constraints or specific run-time properties of the graph assessed via texture measurements.

3.0 A Mediated Approach to Coordination

The constraint relaxation schema defines a family of centralized algorithms where the executing agent has access to the entire constraint graph. In a multiagent environment, the constraint graph is distributed among the agents. No complete representation exists. We identify mediation and negotiation as extremes on a spectrum of conflict recovery techniques.

Negotiation is characterized by an exchange of information among agents so that each can attain a more global view of the interactions and conflicts. Each agent will typically plan local actions in an attempt to avoid conflict with others and achieve some goals. A number of variations on negotiation have been investigated (e.g. [Lesser 81] [Davis 83] [Rosenschein 85] [Conry 86] [Durfee 87] [Sycara 89] [Durfee 91]).

Mediation uses a special agent that gathers information from other agents and forms a global perspective. The mediator attempts to re-form interactions among the agents such that conflicts will minimized and achieve the global and local goals will be achieved. As a starting point, we adopt a mediated approach to conflict management in the supply chain for a number of reasons:

- The supply chain domain is highly-structured and requires explicit conflict recovery techniques only when unexpected events occur.
- A mediator minimizes the conflict recovery knowledge overhead required at each agent.
- The constraint graph upon which relaxation needs to be performed is already partially represented in the existing schedule. The agent responsible for the scheduling is a natural choice as a mediator.
- Conflict recovery in the supply chain is necessary when the commitment graph is found to be infeasible. Given that a schedule forms the core of this graph, we conjecture that conflicts arise for reasons similar to those that lead to scheduling difficulties. Often scheduling difficulties can be traced to a particular scarce resources [Ow 87] [Fox 90] [Sadeh 91] . The work of [Sathi 89] indicates that problems with such keystone elements can be solved better by mediated protocols.

The graph representing the commitments between agents is built by the mediator starting with the current schedule. The mediator requests information from the other agents which is adds to the graph to create the *augmented constraint graph*. The information is composed of newly relevant constraints not represented at the higher level, changes to the higher-level constraints, and the costs of relaxation. Because the mediator requests information from specific agents the communication can be limited to the information bearing directly on the current focus of attention.

In general, the mediator must know where to find the relaxation information. Any model of the supply chain is extensible because additional knowledge is assigned to an agent and the mediator is informed of the agent's abilities. A case in point is the role of the Transportation Manager (TM). We do not include the TM in the example below, however its addition to the supply chain would necessitate no changes in the conflict recovery protocol. In modeling transportation it is necessary to insert transportation activities between production activities at physically disparate locations. The TM has the knowledge about relaxations to transportation activities (e.g. shipping by plane instead of rail may decrease the duration of the activity but increase the cost). The mediator is aware of the location of this knowledge in the supply chain and so will communicate with the TM if a transportation activity is within the focus of the conflict recovery mechanism.

The hierarchical nature of the conflict recovery mechanism is manifest by various levels of mediation. Typically, if a disruptive event occurs within a factory, the factory itself will first attempt to resolve the conflict. This resolution takes the form of a constraint relaxation algorithm on an augmented graph created from communication of agents within the factory. This resolution can only be successful if the commitments internal to the factory can be reconfigured without effecting external commitments. If the factory fails at one level of abstraction, conflict resolution is necessary at the next-highest level. Therefore, though the logistics level of the supply chain is not directly interested in machine breakdown at a particular factory, it may have to deal with conflicts produced by the breakdown.

4.0 Conflict Recovery in the Supply Chain

We now present an example of the communication necessary in reaction to a disruptive event in the supply chain. This example should be viewed as a sketch of the interactions we expect among the agents in the supply chain. Work is progressing on empirical validation of the assumptions underlying the example.

The supply chain is populated by the following agents:[7]

- **Logistics**: responsible for the logistics-level scheduling of activities at each factory. Logistics is also the mediator at this level.
- **Order Acquisition** (OA): responsible for all order-related contact with customers.
- **Resource Manager** (RM): responsible for managing both the consumable resources (e.g. - ordering raw materials) and the usable resources (e.g. - maintenance on machines).
- **Factory Agents**: each factory has an agent that is the factory-level scheduler.
    - Factory$_1$: with machines: M_{11}, M_{12},
    - Factory$_2$: with machines: M_{21}, M_{22},
    - Factory$_3$: with machines: M_{31}, M_{32}.

At the logistics level, the normal mode of execution is the acceptance of a new order and appropriate modification of the existing logistics level schedule. Each factory receives a modified partial schedule and changes its factory level schedule accordingly.

There are three scheduled orders, each with a release date of t_0 and a due date of t_3. Each order consists of three unit-duration activities. Each activity uses one unit-capacity machine. The schedule is shown in Table 1.

Start Times	Machines					
	M_{11}	M_{12}	M_{21}	M_{22}	M_{31}	M_{32}
t_0		A_{11}				A_{13}
t_1	A_{31}	A_{12}	A_{22}		A_{21}	
t_2		A_{23}			A_{32}	
t_3			A_{33}			

TABLE 1. Current Schedule in the Supply Chain Simulation

7. These agents are a subset of those used in our supply chain management simulation.

At time t_{-1}, the customer for O_3 requests that the order be changed requiring that A_{13} use M_{12}. Logistics investigates the options of moving activities to other compatible machines or delaying the delivery of one or more of the products. A representation of the communication that takes place for the former option is as follows:

- Logistics to RM: "What machines can be substituted for M_{12}?"
- RM to Logistics: "M_{22}, but quality will decrease."[8]
- Logistics to OA: "What is the cost of the reduced quality on O_2 and O_1?"
- OA to Logistics: "3 each."

For the latter option, Logistics must find the cost of late delivery of each order:

- Logistics to OA: "What is the cost of late delivery on each order?"
- OA to Logistics: "O_1: 5 per time unit, latest acceptable delivery is t_6."
- OA to Logistics: "O_2: 5 per time unit, latest acceptable delivery is t_6."
- OA to Logistics: "O_3: 2 per time unit, latest acceptable delivery is t_6."

The information from the other agents allows the mediator to form the augmented constraint graph and execute a relaxation algorithm on it. We expect the graph to somewhat similar to the constraint graphs in the optimization problems investigated in [Sadeh 91] . Instantiations of the relaxation schema are executed on this augmented graph by the mediator.

5.0 Conclusion

We have presented a mediated approach to conflict resolution using constraint relaxation. Having modeled the interactions of the agents as a commitment/constraint graph, deviations from the normal actions of the agents may produce an overconstrained situation. A mediator gathers information in the form of variables, constraints, and relaxation costs and executes a constraint relaxation algorithm on the augmented graph. The result is a near optimal modification to the constraints to re-establish a feasible graph.

The use of constraint relaxation as a conflict recovery operator builds on the power and expressivity of the constraint model. With the appropriate modeling of problems in the constraint formalism, properties of the structure of the problem can be assessed via texture measurements. The experimental results on PCSPs and schedule optimization highlight the generality and flexibility of the constraint relaxation schema we have developed. Within the supply chain simulation, empirical investigation of the properties of commitment graphs and of appropriate texture measurements will provide a basis for conflict management via constraint relaxation in other multiagent domains.

Mediation is one end of a spectrum of conflict management techniques and constitutes our first step in distributing the constraint relaxation algorithm. The algorithm itself is still centralized, as it is executed by a single agent on a graph produced from information gathered from other agents. Further work will attempt to move toward a more distributed approach by the combination of negotiation and mediation based on problem properties. The mediated approach will provide data against which more distributed approaches can be compared.

6.0 Acknowledgments

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

7.0 References

[Beck 94] Beck, J.C. *A Schema for Constraint Relaxation with Instantiations for Partial Constraint Satisfaction and Schedule Optimization.* Master's thesis, University of Toronto, 1994. To Appear.

8. There will need to be a quantification of quality for such a response to be meaningful. Work is progressing on such a theory of quality that will be a part of a shared ontological knowledge base [Kim 94] .

[Conry 86] Conry, S.E., Meyer, R.A., Lesser, V.R. *Multistage Negotiation in Distributed Planning.* Technical Report 86-67, Department of Computer and Information Science, University of Massachusetts, December, 1986.

[Davis 83] Davis, R. and Smith, R.G. Negotiation as a Metaphor for Distributed Problem Solving. *Artificial Intelligence.* 20(1):63-109, 1983.

[Durfee 91] Durfee, E.H. and Montgomery, T.A. Coordination as Distributed Search in a Hierarchical Behavior Space. *IEEE Transactions on Systems, Man, and Cybernetics.* SMC-21(6):1361-1378, 1991.

[Durfee 87] Durfee, E.H. and Lesser, V.R. Using Partial Global Plans to Coordinate Distributed Problem Solvers. *Proceedings of IJCAI-87*, pages 875-883. 1987.

[Fox 90] Fox, M.S. and Sadeh, N. Why Is Scheduling Difficult? A CSP Perspective. *Proceedings of the Ninth European Conference on Artificial Intelligence.* 1990.

[Fox 89] Fox, M.S., Sadeh, N., and Baykan, C. Constrained Heuristic Search. *Proceedings of IJCAI-89.* 1989.

[Freuder 92] Freuder, E. and Wallace, R. Partial Constraint Satisfaction. *Artificial Intelligence.* 58:21-70, 1992.

[Kim 94] Kim, H.K. and Fox, M.S. Formal Models of Quality and ISO 9000 Compliance: An Information Systems Approach. *48th Annual Quality Congress of the American Society of Quality Control.* 1994.

[Lesser 81] Lesser, V.R and Corkill, D.D. Functionally Accurate, Cooperative Distributed Systems. *IEEE Transactions on Systems, Man, and Cybernetics.* SMC-11(1):81-96, January, 1981.

[Nagel 91] Nagel, R.N. et al. *21st Century Manufacturing Enterprise Strategy: An Industry Led View.* Technical Report, Iacocca Institute, Lehigh University, Bethlehem PA, 1991.

[Ow 87] Ow, P.S. and Smith, S.F. Viewing Scheduling as an Opportunistic Problem-Solving Process. *Annals of Operations Research: Approaches to Intelligent Decision Support.* In Jeroslow, R. G., Balzer Scientific Publishing Co., 1987.

[Rosenschein 85] Rosenschein, J.S. and Genesereth, M.R. Deals Among Rational Agents. *Proceedings of IJCAI-85*, pages 91-99. 1985.

[Sadeh 94] Sadeh, N. *personal communication.*, 1994.

[Sadeh 91] Sadeh, N. *Lookahead Techniques for Micro-Opportunistic Job Shop Scheduling.* PhD thesis, Carnegie Mellon University, 1991. CMU-CS-91-102.

[Sadeh 89] Sadeh, N. and Fox, M.S. *Preference Propagation in Temporal/Capacity Constraint Graphs.* Technical Report CMU-RI-TR-89-2, The Robotics Institute, Carnegie Mellon University, January, 1989.

[Sathi 89] Sathi, A. and Fox, M.S. Constraint-Directed Negotiation of Resource Reallocations. Volume 2. *Distributed Artificial Intelligence.* In Michael N. Huhns and Les Gasser, Pitman Publishing & Morgan Kaufmann Publishers, 1989, pages 163-193, Chapter 8.

[Sycara 91] Sycara, K., Roth, S., Sadeh, N., and Fox, M. Distributed Constrained Heuristic Search. *IEEE Transactions on Systems, Man, and Cybernetics.* SMC-21(6):1446-1461, 1991.

[Sycara 89] Sycara, K.P. Argumentation: Planning Other Agents' Plans. *Proceedings of IJCAI-89*, pages 517-523. 1989.