Managing Numeric and Symbolic information for Scheduling and Planning

Malek Mouhoub
Department of Math & Computer Science, University of Lethbridge
4401 University Drive,
Lethbridge, Alberta, Canada, T1K 3M4
mouhoub@cs.uleth.ca

Abstract

In this paper we present two methods for managing both the numeric and the symbolic aspects of time. The first method is used to check for the consistency of a problem involving time constraints, and returns a coherent scenario, called also solution of the problem, if the latter is consistent. The second method is of interest when a solution of a problem involving time constraints should be returned within a given deadline. This is the case, for example, of real time applications where a complete solution cannot be obtained without missing the deadline. The method returns then an incomplete solution which has the guaranty to be optimal (solution violating the minimum number of constraints). Both methods are based on constraint satisfaction techniques.

Keywords: temporal reasoning, partial constraint satisfaction techniques, any-time algorithms, planning, scheduling.

1 Introduction

Time is an important parameter of any intelligent knowledge representation. There is thus a rising need for reasoning about time in various applications such as planning, scheduling, process control, decision support systems, molecular biology and natural language processing. Two main aspects of time are largely studied : the numeric and the symbolic aspect of time. The separation between the two aspects does not exist in the real world; In our daily life activities, for example, we combine the two type of information to describe certain situations. In the case of scheduling problems, we can have qualitative information such as the ordering between tasks and quantitative information describing the temporal windows of the tasks i.e earliest starting time, latest end time and the duration of each task.

Research works addressing the problem of handling both metric and symbolic constraints have already been reported. Meiri [Mei96] has proposed a model based on a single network (time map) managing both constraints : metric constraints that restrict the distance between time points, and symbolic constraints that specify the relative position between temporal objects (either points or intervals). Kautz and Ladkin [KL91] have proposed a model allowing for the representation and processing of metric temporal information in the form of a system of simple linear inequalities to encode metric relations between time points, and systems of binary constraints in Allen's qualitative temporal calculus to encode qualitative relations between time points. In the Kautz and Ladkin approach, both kind of constraints are independently processed in separate networks.

In order to manage numeric and symbolic information, we have proposed a model, TemPro[MMH98], which extends the Interval Algebra[All83] to handle numeric information. The algorithmic approach is that of constraint satisfaction over discrete domains[Mon74] [Mac77][HE80]. Indeed, using a discrete representation of time allows a translation of a problem involving time information into a constraint satisfaction problem that can easily be solved using constraint satisfaction techniques.

At the representation level, our approach is close to that of Kautz and Ladkin, but it differs in the way metric constraints are defined. Quantitative constraints are represented by linear inequalities in the Kautz and Ladkin approach, while they are modeled in our representation by a set of numeric intervals that each temporal event can take. Thus, we come out with a uniform language at the numerical and symbolic level . In the approach of Kautz and Ladkin, the propagation is performed separately in both metric and qualitative networks. Translation procedures are then used to update the metric constraints from the new Allen constraints and vice versa. This process is repeated until no new statements can be derived. As we will see in the next sections, in our approach the process to compute Allen and metric constraints

network is performed in one pass. A backtracking algorithm is then used to search for a possible numeric solution.

In this paper we present two methods for managing numeric and symbolic time information. After representing a given problem involving time information by our model TemPro, the first method is used to check for the consistency of the problem, and returns a coherent scenario, called also solution of the problem, if the latter is consistent. The method is based on constraint satisfaction techniques that we have improved using time properties.

The goal of the second method is to deal with real time applications or those applications where a complete solution cannot be obtained. To reach this goal we have modified the constraint satisfaction techniques used in the first method so that they will be able to solve temporal problems by giving a solution with a quality depending on the time allocated for computation.

In the following section, we will present through an example, the different components of the model TemPro. The description of the two methods is then presented in section 3. Section 4 concerns scheduling problems. Some experiments evaluating the techniques we propose are reported in section 5. Concluding remarks and possible perspectives of our work are presented in section 6.

2 Knowledge Representation

Consider the following typical temporal reasoning problem :

> *John, Mike and Lisa work for a company in Calgary. It takes John* **20 minutes***, Mike* **15 minutes** *and Lisa* **35 minutes** *to get to work. Every day, John left home* **between 7:10 and 7:22***. Mike arrives at work* **between 7:50 and 7:55** *and Lisa arrives at work* **between 7:50 and 8***. We also know that John and Mike* **meet** *on their way to work, that Mike arrives to work* **before** *Lisa and that Lisa and John go to work* **at the same time***.*

The above story includes numeric and qualitative information (words in boldface). There is three main events : John, Mike and Lisa are going to work respectively. Some numeric constraints specify the duration of the different events, e.g *15 minutes is the duration of Mike's event*. Other numeric constraints describe the temporal windows in which the different events occur. And finally, symbolic constraints state the relative positions between events e.g *John and Mike meet on their way to work*.

Given these kind of information, our main goal is to represent and reason about such knowledge and answer queries such as : "is the above problem consistent ?", "what are the possible times at which Lisa arrived at work ?".
To reach this goal, we first translate the temporal problem into a constraint satisfaction problem. Our model TemPro is used for this purpose. Constraint satisfaction techniques are then used to check for the consistency and look for possible solutions. In the following, we will see the different objects of our model TemPro.

2.1 Events

In TemPro, temporal objects are called events. Events have a uniform reified representation made up of a proposition and its temporal qualification : $Evt = OCCUR(p, I)$ defined by Allen [All83] and denoting the fact that the proposition p occurred over the interval I. For the sake of notation simplicity, an event is used in this paper to denote its temporal qualification. According to our model, in our example we have three main events : J, L and M representing the fact that John, Lisa and Mike are going to work respectively.

2.2 Qualitative Constraints

Qualitative constraints specify the relative temporal position of an event with respect to other events. The qualitative constraint between two events ev_1 and ev_2 can take the following forms : $I_1\ r_1\ r_2 \ldots r_n\ I_2$ where $ev_1 = OCCUR(p, I_1)$, $ev_2 = OCCUR(p, I_2)$ and each of the r_i 's is one of the thirteen Allen primitives : $Precedes$, $during$, $overlaps$, $meets$, $starts$, $finishes$ noted respectively P, D, O, M, S and F; their converses $P^{\smile}$, $D^{\smile}$, $O^{\smile}$, $M^{\smile}$, $S^{\smile}$ and $F^{\smile}$; and the equality relation E.

From our example, we can extract the following qualitative constraints :

- M $E\,O\,O^{\smile}\,S\,S^{\smile}\,F\,F^{\smile}\,D\,D^{\smile}$ J : representing the fact "*John and Mike meet at the traffic light.*"
- M $D\,O\,S\,P$ L : representing the fact "*Mike arrives to work before Lisa.*"
- L $E\,S\,S^{\smile}$ J : representing the information "*Lisa and John go to work at the same time.*"

2.3 Quantitative Constraints

Additional information about an event can be expressed, thus restricting its temporal qualification to belong to a given SOPO, i.e., the Sets Of Possible Occurrences where the given event can take place. In our case, a SOPO is restricted to be a finite set of intervals with constant duration. We represent a SOPO by the fourfold $[begintime, endtime, duration, step]$ where $begintime$ and $endtime$ are respectively the earliest begin time and the latest end time of the corresponding event, $duration$ is the duration of the event and $step$ defines the distance between the starting time of two adjacent intervals

within the SOPO. Thus, if e_i is an event numerically constrained by the SOPO $[inf_i, sup_i, d_i, p_i]$, then the set of possible occurrences of e_i is defined as :
$I = \{occ_j \mid begin(occ_j) = inf_i + k * p_i, end(occ_j) = begin(occ_j) + d_i * p_i, end(occ_j) <= sup_i, k \in [0, \frac{sup_i - inf_i}{p_i} - d_i] \cap \aleph\}$.

$begin$ and end are functions on intervals that return the begin and the end points of a given interval, respectively.

In our example, we have the following quantitative constraint. The step of discretization is equal to 1min (see also figure 1) :

- $J = (P, I) \wedge I \in [15, 42, 20, 1] = \{(15, 32), (16, 33), \ldots, (22, 42)\}$
- $M = (P', I') \wedge I' \in [35, 55, 15, 1] = \{(35, 50), (36, 51), \ldots, (40, 55)\}$
- $L = (P'', I'') \wedge I'' \in [20, 60, 30, 1] = \{(20, 50), (21, 51), \ldots, (30, 60)\}$

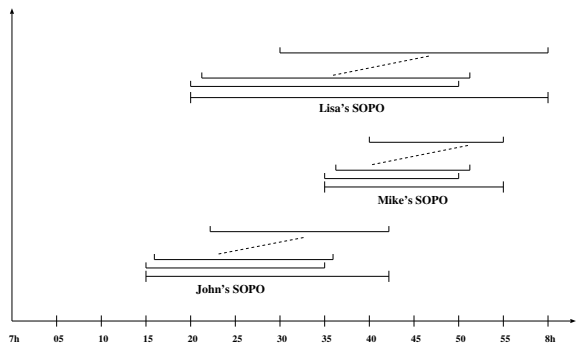


Figure 1. Numeric constraints (SOPOs).

3 Solving Qualitative and Metric Temporal Constraint Problems

3.1 TemPro based Temporal Constraint Satisfaction

In the previous section, we have shown how both qualitative and metric constraints are represented in TemPro. TemPro transforms a temporal problem involving numeric and symbolic information into a constraint satisfaction problem involving a set of variables $\{EV_1, \ldots, EV_n\}$, each defined on a discrete domain D_i standing for the set of possible occurrences in which the corresponding event can hold, and a set of binary constraints, each representing a qualitative disjunctive relation between a pair of events. Using TemPro, our example can be represented by the following temporal constraint network (see first graph of figure 2).

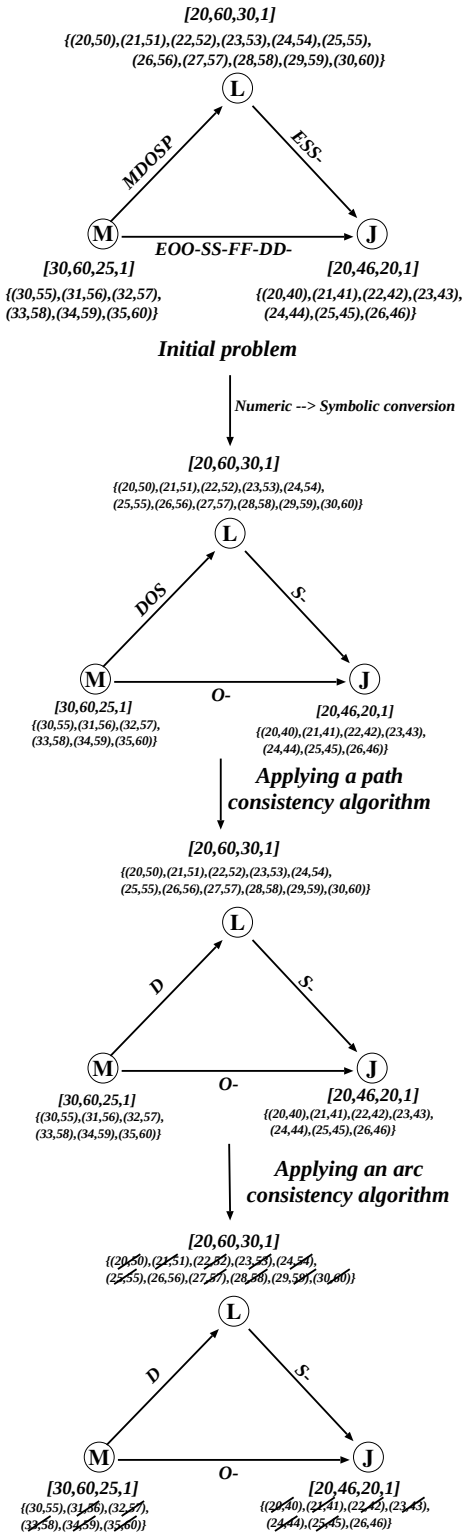


Figure 2. Numeric → symbolic conversion + pre-processing.

3.2 Finding a single solution

Given a TemPro constraint-based network (see figure 2), one important task of reasoning is to determine the consistency of the network, and therefore to find a solution of this network if the latter is consistent. Since we are dealing with a constraint satisfaction problem, deciding consistency is NP-hard. In order to overcome these difficulties, we propose a search algorithm based on local consistency techniques. Indeed, these algorithms transform the network of constraints into an equivalent and simpler one by removing, from the domain of each variable, some values that cannot belong to any global solution. A k-consistency algorithm removes all inconsistencies involving all subsets of k variables belonging to N. For $k = 2$ and $k = 3$, the solutions are called arc and path consistent respectively. The k-consistency problem is polynomial in time $O(n^k)$, where n is the number of variables. A k-consistency algorithm does not solve the constraint satisfaction problem, but simplifies it. Due to the incompleteness of constraint propagation, in the general case, search is necessary to solve a CSP problem, even to check if a single solution exists. Our resolution algorithm is described by the following steps (see also figure 2) :

Numeric → Symbolic conversion : *From the numeric information, we can extract the corresponding symbolic relation. An intersection of this relation with the given qualitative information will reduce the size of the latter which simplifies the size of the original problem.*

Pre-processing : *Perform some local consistency inferences (arc and path consistency algorithms) on the numeric and symbolic levels in order to reduce the size of the problem. Figure 2 presents the temporal network of our example after applying the pre-processing step i.e path consistency on the symbolic relations and arc consistency on the numeric constraints (SOPOs).*

Backtrack search : *Perform a backtrack search algorithm in order to look for a single solution. This can be performed as follows :*

Choose a node and instantiate the corresponding variable to an occurrence belonging to its domain. Discard from its domain the remaining values and run some consistency algorithms in order to restore consistency. If the network succeeds, fix an occurrence on another variable and run again the local consistency algorithms until each occurrence is fixed on the domain of each variable of the network. We obtain a solution corresponding to the set of the occurrences fixed on the domain of each variable. If the network does not succeed at some point, backtrack and choose another occurrence on the domain of the last selected variable.

In a previous work [MMH98], we have experimentally compared different local consistency algorithms performed at the symbolic and numeric levels in the pre-processing and search phases. We concluded the following :

- path consistency is very expensive to find a single numeric solution, but is useful to detect inconsistencies of TemPro networks,
- arc consistency, in a limited form (following the principle of the forward-check) during the numeric solution search, outperforms other approaches,
- The arc consistency algorithms AC-3 and AC-7 [BFR95] have complementary application domain depending on the complexity of the problem, as indicated in the following table :

	under constrained problems	middle and over constrained problems	scheduling problems
AC3	pre-processing and backtrack search	backtrack search phase	pre-processing and backtrack search
AC7	–	pre-processing phase	–

3.3 Solving temporal constraint problems in real time

In the real world, a main challenge for temporal constraint problems like scheduling and planning applications is the availability to run on-line, that is to give a solution quickly enough so that the system can use it without missing the deadline of the first events. However, most of the temporal problems are NP-Hard. Heuristic and approximation algorithms are in general used for large problems[SML92][SK93] and, obviously, are of interest when they provide near optimal solutions with a polynomial computational complexity. In our work, we have developed a complete method, which, in contrast to heuristic methods, can provide solutions to a given problem that are guaranteed to be optimal. In our work, optimality refers to the number of satisfied constraints. The method is based on partial constraint satisfaction techniques[FW92][Wal95], capable of solving temporal constraint problems by giving a solution with a quality depending on the time allocated for computation. It consists in using a branch and bound variant of the search algorithm in order to satisfy a maximum number of constraints. We have then transformed the propagation techniques we have seen in the previous section, as follows :

1. In the pre-processing phase of the resolution method, either than performing arc consistency algorithms to remove some values that do not belong to any solution, we use direct arc consistency algorithms[DP88][Wal95] to count the number of inconsistencies counts associated with each value (number of domains that offer no support for

that values). Note that, with direct arc consistency algorithms, checking is unidirectional so inconsistency counts are non-redundant.
In order to perform direct consistency checking, a variable ordering should be first established. The ordering heuristic we use consists in sorting variables by decreasing number of constraints shared with those already instantiated.

2. In the search phase, a cost function corresponding to the number of violated constraints is associated with each path of the search tree. The branch and bound algorithm starts by setting an upper bound value on the cost function. Search is then performed down on each path until all variables are instantiated and in this case a new lower bound is found, or when a partial assignment of values to variables is at least as great as the actual lowest cost.

The following figure (see figure 3) describe our solving system handling both methods. The user has to input the temporal problem in the form of a list of numeric and symbolic information. These information will be then translated into a temporal constraint network. The numeric $\rightarrow$ symbolic conversion will then be performed to reduce the size of the problem. This phase is also used as a preliminary test to check if the problem is inconsistent. If it is not the case, the user will have to choose between two options : run the system in real time or look for a complete solution. If the user chooses the real time option, the solver will start looking for a possible solution and the user can interrupt it at any time to get a solution with a certain quality (the quality of a solution corresponds to the number of satisfied constraints). If the second option is chosen, the solver will run until it finds a complete solution or returns that the problem is inconsistent.

4 Solving scheduling problems with non pre-emptive tasks

Consider the following scheduling problem :

Three tasks T_1, T_2 and T_3 are processed by a mono processor machine. A task T_4 must be processed before T_1 and T_2. Each task is characterized by a processing time, an earliest begin time and a latest end time. We assume that the tasks are non pre-emptives.

The above example can easily be translated into a TemPro network. Indeed, the different tasks can be represented by temporal events where the SOPOs define the numeric temporal information of the tasks and the qualitative constraints indicate the relative temporal position of a task compared to another. For example, $T_1 \quad P \vee P^{\smile} \quad T_2$ represent

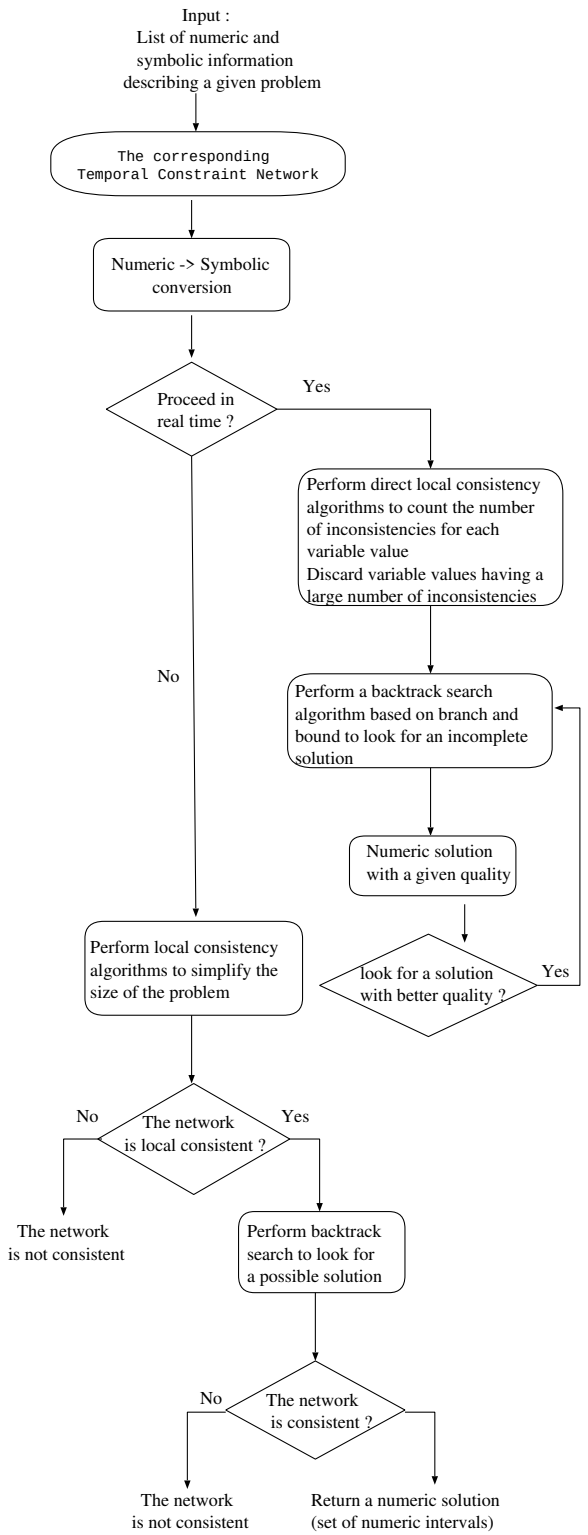
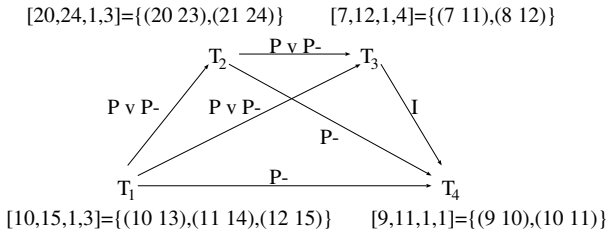


Figure 3. Solving system based on TemPro.

Tasks	Earliest begin time	Latest end time	duration
T_1	10	15	3
T_2	20	24	3
T_3	7	12	4
T_4	9	11	1

Table 1. Numeric information of the different tasks

the fact that the tasks T_1 and T_2 are executed in mutual exclusion. Figure 4 describes the representation of the above example by the model TemPro. Note that, since there is no constraint between task T_3 and T_4, we represent the corresponding relation by the universal relation I which is the disjunction of all possible basic relations between time intervals.



I : The universal relation(disjunction of the 13 basic Allen relations).
P : Precedes, P- : precedes inverse.

Figure 4. TemPro representation of a scheduling problem.

5 Experimentation

In order to evaluate our algorithm, we have performed some experiments on temporal constraint problems randomly generated with complete solutions. The experiments were performed on a SUN SPARC Ultra 5 station. All the procedures are coded in C/C++. The temporal problems are randomly generated as follows :

Randomly pick n (n is the number of variables of the problem to generate) pairs (x, y) of integers such that $x < y$ and $x, y \in [0, \ldots, Horizon]$ ($Horizon$ is the parameter before which all events must be processed). This set of n pairs forms the initial solution where each pair corresponds to a time interval.

Metric constraints are generated as follows:

For each interval (x, y) randomly pick an interval contained within $[0..Horizon]$ and containing the interval (x, y). These newly generated intervals define the SOPO of the corresponding variable.

Qualitative constraints are obtained as follows :

Compute the basic Interval-Interval relations that can hold between each interval pair of the initial solution. Add to each relation a random number in the interval $[0, Nr]$ ($1 < Nr \leq 13$) of chosen basic Interval-Interval relations.

The generated problems can be characterized by their tightness, which can be measured, as shown in [SF94], using the following definition :

> *The tightness of a CSP problem is the fraction of all possible pairs of values from the domain of two variables that are not allowed by the constraint.*

The tightness depends in our case on the parameters $Horizon, Nr$ and the density of the problem.

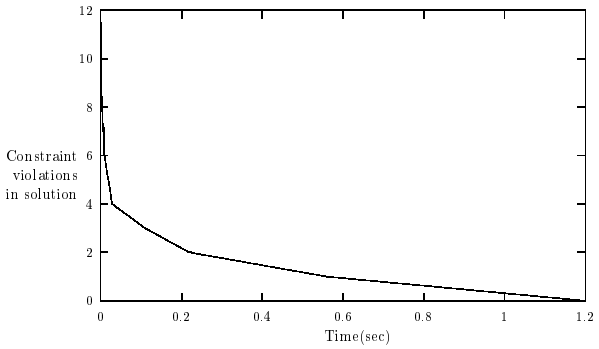


Figure 5. Averaged anytime curve. 100-variable problems, tightness=0.10

The "anytime curve" (see figure 5) is based on the average number of constraint violations found after each period of time. The curve reports tests performed on 100 problems, of size 100 each, with complete solutions.

For small size problems, our method provides interesting results. However for large problems, the algorithm becomes impracticable and heuristic methods are then the procedures of choice. In this case, our method can be used to evaluate the goodness of results returned by heuristic methods.

6 Conclusion and Future Work

In this paper we have presented a system that handles temporal problems involving both numeric and symbolic time information. The main task of the system is to check and maintain the consistency of a given temporal problem according to the different time constraints. This is very important for many industrial applications such as scheduling, planning and natural language processing. The system is also capable to deal with real time applications, that is

to provide a incomplete solution without missing a given deadline. The solution has then the guaranty to be optimal (violates the minimum number of constraints).

In the near future, we intend to integrate our model TemPro into a planning system. Indeed, Planning systems based on the state space approach, for example, use a sequential notion of time (systems like TWEAK[Cha87] and SNLP[MR91] are based on a ordering relation between operators). Our goal is to enrich the time representation of the planning system especially by handling numeric and symbolic constraints. A problem to plan will include, in addition to the implicit temporal relations (orderings generated during the planning process to solve some resource conflicts), a set of explicit temporal constraints. One of the main roles of the planner is to maintain, during the construction of the plan, the consistency of the latter given the temporal constraints. The resolution techniques we have developed will then be used to perform the temporal constraint propagation.

Another perspective consists in solving temporal constraint problems in a dynamic environment. We may, for example, add new information or relax some constraints when there are no more solutions. In those cases we need to check if there still exist solutions to the problem every time a constraint has been added or removed. Adding temporal constraints can easily be handled by TemPro, we have just to put in this case the new constraint in the lists of constraints to be checked. However, constraint relaxation can not be handled by our techniques. Indeed, when we remove a constraint, our algorithms cannot find which value, that has been already removed, must be put back and which one must not. We must then transform our satisfaction techniques so that they will be able to save the reason why a certain value has been removed.

We also intend to extend our model TemPro in order to manage incomplete information, that is :

1. To handle estimated duration and temporal windows bounds. If we have to plan different tasks in an operating system for example, the corresponding processes will have an estimated duration depending on certain parameters. In this case we have to think of a new method of propagating estimated constraints,

2. To manage assumptions in the case where a given time information is incomplete. Managing assumptions can be handled by an assumptions truth maintenance systems (ATMS)[dK86], however we may also use constraint satisfaction techniques in this case. Indeed, ATMS and CSP techniques embody the same computational trade-offs. CSP local consistency algorithms, for example, are motivated by exactly the same set of problems solving concerns as the ATMS algorithms. Also, most of the CSP local consistency algorithms themselves have almost identical analogs in the ATMS framework.

References

[All83] J.F. Allen. Maintaining knowledge about temporal intervals. *CACM*, 26(11):832–843, 1983.

[BFR95] C. Bessière, E. Freuder, and J.C. Regin. Using inference to reduce arc consistency computation. In *IJCAI'95*, pages 592–598, Montréal, Canada, 1995.

[Cha87] D. Chapman. Planning for conjunctive goals. *Artificial Intelligence.*, 32:333–377, 1987.

[dK86] J. de Kleer. An assumption-based truth maintenance system. *Artificial Intelligence*, 28:127–162, 1986.

[DP88] R. Dechter and J. Peal. Network-based heuristics for constraint satisfaction problems. *Artificial Intelligence*, 34:1–38, 1988.

[FW92] E. C. Freuder and R. J. Wallace. Partial constraint satisfaction. *Artificial Intelligence*, 58:21–70, 1992.

[HE80] R.M. Haralick and G.L. Elliott. Increasing tree search efficiency for Constraint Satisfaction Problems. *Artificial Intelligence*, 14:263–313, 1980.

[KL91] H.A. Kautz and P.B. Ladkin. Integrating metric and qualitative temporal reasoning. In *AAAI'91*, pages 241–246, Anaheim, CA, 1991.

[Mac77] A. K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8:99–118, 1977.

[Mei96] I. Meiri. Combining qualitative and quantitative constraints in temporal reasoning. *Artificial Intelligence*, 87:343–385, 1996.

[MMH98] F. Charpillet Malek Mouhoub and J.P. Haton. Experimental analysis of numeric and symbolic constraint satisfaction techniques for temporal reasoning. *Constraints: An International Journal*, 2:151–164, June 1998.

[Mon74] U. Montanari. Fundamental properties and applications to picture processing. *Information Sciences*, 7:95–132, 1974.

[MR91] D. McAllester and D. Rosenblitt. Systematic nonlinear planning. In *AAAI-91*, pages 634–639, 1991.

[SF94] D. Sabin and E. C. Freuder. Contradicting conventional wisdom in constraint satisfaction. In *Proc. 11th ECAI*, pages 125–129, Amsterdam, Hollande, 1994.

[SK93] B. Selman and H. A. Kautz. An empirical study of greedy local search for satisfiability testing. In *AAAI'93*, pages 46–51, 1993.

[SML92] A. B. Philips S. Minton, M. D. Johnston and P. Laird. Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence*, 58:161–205, 1992.

[Wal95] R. J. Wallace. Partial constraint satisfaction. *Lecture Notes in Computer Science*, 923:121–138, 1995.