Hybrid Temporal Reasoning for Planning and Scheduling

Silvana Badaloni

Dept. of Electronics and Computer Science
Padua, Italy 35100

Marina Berati

CSELT
Turin, Italy 10148

Abstract

This paper address the problem of representing heterogeneous temporal information in a uniform framework. Metric information relative to intervals is combined with qualitative information in a homogeneous representation based on a temporal constraint network. We illustrate the properties of the new sub-algebra (called IDSA), the algorithms used to propagate temporal information and their complexity.

1 Introduction

In this paper, we present a new formalism, called IDSA (Interval Distance Sub Algebra) that allows us to represent both qualitative and quantitative constraints on temporal intervals. Our main commitments are:

(i). to blend metric information relative to intervals to qualitative information in a homogeneous representation based on intervals as elementary entities

(ii). to augment the expressive power of the system compared with Interval Algebra [1] and to Distance Algebra [9]

(iii). to represent disjunction of qualitative relations (e.g. interval I1 is before or after interval I2) since we deal with a planning/scheduling problem

(iv). to represent disjunction of metric and qualitative constraints since we deal with possibly uncertain knowledge

(v). to utilize complete algorithms for propagating local constraints.

The temporal model we consider is the interval based logic proposed by Allen which we have extended by including metric information in order to build a temporal planner capable of generating plans involving deadlines and quantitative temporal constraints [4], [5], [6].
The paper is organised as follows: in section 2 we introduce the temporal representation and explain which kind of temporal information it can represent; in section 3 we analyse the properties of the IDSA subalgebra; in section 4 we examine the algorithms used in our system to perform the typical operations that can be done over a temporal network; in section 6 we compare our formalism with other hybrid approaches.

2 The temporal representation

The basic temporal entity is time interval. A given fact that occurs or holds over a certain time period, is a temporally qualified fact, that is, it has a temporal qualification, the interval during which the fact takes place. In the line of reifed logics, it is defined by a pair $\langle predicate, interval \rangle$. For example, a *property* can be defined according to:

$$holds(P, I) \leftrightarrow (\forall t \; . \; in(t, I) \rightarrow holds(P, I))$$

where the temporal relantionship $in = \{s, d, f, e\}$: if a property P holds over an interval I, it holds over all sub-intervals of I. According to Allen, the temporally qualified facts can be properties, processes and events. Indeed, the notion of processes is not particularly useful in practical applications; furthermore, as noticed by [14], its definition is rather controversial.
Intervals and constraints are represented in a temporal constraint network: nodes represent temporal intervals, the directed arcs are labelled by unary and binary temporal constraints and self-loops by unary constraints. Let us now define unary and binary constraints.
Definition. A *unary constraint* is defined as the disjunction of possible durations for an interval:

$$D_1 \vee D_2 \vee ... \vee D_n$$

where D_i represents the range of values of the interval duration, in the form $d(X, Y)$, where $X \in \Re$, $Y \in \Re$ and $X \leq Y$: $d(X, Y)$ may be open or closed at one or both of its bounds. Obviously, the values $-\infty$ (minf) and $+\infty$ (pinf) are allowed.
The set of durations of an interval defining the constraint in a disjunctive form is represented as a list. Thus, to assert that an interval A can assume a duration included between 4 and 10, or it can last more than 20 (referred to a given time scale), we consider the unary constraint relative to A:

$$[d(4, 10), d(20, pinf)]$$

0-8186-7528/96 $5.00

When introducing a unary constraint into the network, the system translates it into a *canonical form*, such that all the durations defined in the list are mutually disjoint.
Let A and B be two intervals and A_1, A_2, B_1 and B_2 their starting and ending points, respectively.
Definition. A *binary constraint* between two intervals A and B is defined as a disjunction of binary relations that can hold between those intervals:

$$R_1 \vee R_2 \vee ... \vee R_n$$

where $R_i \in$ IDSA (Interval Distance Sub-Algebra), for $i = 1..n$. The binary relation R_i between A and B is an ordered tuple of four intervals belonging to the real axis. It defines the mutual position of the intervals A and B, blending together the quantitative and qualitative temporal information.
Let the intervals D_{11}, D_{12}, D_{21} and D_{22} (open, closed or semi-open) the distances $B_1 - A_1$, $B_2 - A_1$, $B_1 - A_2$ and $B_2 - A_2$ respectively.
Definition. Any two intervals A and B satisfy a binary relation R_i if the distances D_{11}, D_{12}, D_{21} and D_{22} belong to the four intervals composing the tuple R_i.
Since uncertainty is allowed, each distance may have a range of variability defined by a lower and an upper bound: it follows that $D_{ij} = d_{ij}(L_{ij}, U_{ij})$ where L_{ij} and U_{ij} are the lower and the upper bounds respectively, for $i, j = 1, 2$; obviously, if the distance is exactly known then L_{ij} and U_{ij} coincide.
Let us now see how, starting from the available temporal information (both qualitative and quantatitive), a binary relation is computed according to the above definition. Generally, we may know either only a qualitative relation RQ between A and B (e.g. A before B), or only some distances between the four endpoints (when a distance is left undefined, it is intended to vary from $-\infty$ to $+\infty$), or, finally, RQ plus some or all distances. This knowledge can be represented as a relation R_{input}:

$$R_{input} = [\ RQ,\ D_{11}^m,\ D_{12}^m,\ D_{21}^m,\ D_{22}^m]$$

where the index m indicates metric information only. It is not necessary to specify all five elements of R_{input}, but only a subset of them is sufficient. The qualitative relation $RQ \in$ SAc where SAc is the pointizable sub algebra of Interval Algebra [15], [16], that is composed of those relations of the interval algebra that can be translated into a conjunction of relations of PAc (continuous point algebra [17]) between the beginning and the ending points of the intervals (e.g. $before \vee overlaps$ is allowed, but $before \vee after$ is not).
To translate such a metric-qualitative relation into an IDSA relation, RQ is translated into a tuple of four distances. For example, the relation $precedes = before \vee meets$ is translated into the tuple:

$$(d_{11}(o(0), +\infty), d_{12}(o(0), +\infty),$$

$$d_{21}(0, +\infty), d_{22}(o(0), +\infty))$$

where o(X) means that the extreme X is open. The binary relation R_i is then computed by intersection of this tuple with the metric one. The tuple thus obtained is an element of IDSA sub-algebra.
In this way, temporal information of different types (qualitative and quantitative) is fused into a single one (only quantitative), that is the tuple of the four distances between the extremes of the constrained intervals. Then, in the internal representation of the temporal reasoner based on the IDSA algebra, computations are performed on metric information only. In fact, when the four distances are specified, the qualitative relation RQ is completely defined, since it can be inferred from the quantitative information (i.e. the IDSA relation).
The results obtained by applying the constraint propagation algorithms are presented in output utilizing the same input format, that is, specifying explicitly both the qualitative relation RQ and the four distances that compose the quantitative relation between intervals. As previously noted, the qualitative relation is redundant, but it is convenient, for many purposes, to render it explicit.
Let us give an example. Suppose we consider two intervals A and B, the duration of A varying from 5 to 6, $d(5, 6)$, and the duration of B varying from 15 to 20, $d(15, 20)$. We know that A is contained in B, and the distance between the ending point of A and the ending point of B is $d_{22}(0, 3)$. To express this information, we must consider two unary relations defining the unary constraints (durations) for the intervals A and B, and a binary constraint, made up of a single binary relation, indicating the relationship between A and B. Thus, the unary constraints are $d(5, 6)$ for A and $d(15, 20)$ for B. The binary relation has the input format:

$$R_{input} = [\ in,\ d_{22}(0, 3)]$$

The corresponding IDSA relation is expressed by the tuple:

$$\begin{aligned} R_i \ = \ & (\ d_{11}(minf, 0),\ d_{12}(o(0), pinf), \\ & d_{21}(minf, o(0)),\ d_{22}(0, 3)) \end{aligned}$$

where o(X) means that the extreme X is open. If an algorithm for propagating the temporal information is executed, so that the unary constraints are taken into account to strengthen the binary constraint, the relation between A and B, in its output format, becomes:

$$\begin{aligned} R_{output} \ = \ & [\ in,\ d_{11}(-15, -6),\ d12(5, 9), \\ & d21(-20, -12),\ d22(0, 3)] \end{aligned}$$

In our representation of temporal information, the elementary entity is still the interval, because unary and binary constraints refer to intervals and nodes in the network represent intervals, even if, in order to specify quantitative information, we refer, on an internal level, to the extremes of intervals, i.e. to time points. Therefore, it is possible to blend metric information relative to intervals with qualitative information in a homogeneous way thus augmenting the expressive power of the system compared with the IA and to the Distance Algebra.

3 Properties of IDSA

The time needed to propagate information over the network, to infer new constraints, is closely dependent on the number of relations per arc. Thus, it is important to keep the number of disjunctions for each constraint as small as possible, so as to reduce computational effort when performing operations on the network. To this aim, let us give the following definitions.

Given two relations R_i and R_j belonging to IDSA, where $R_i = [RQ_i, D^m_{11i}, D^m_{12i}, D^m_{21i}, D^m_{22i})]$ and $R_j = [RQ_j, D^m_{11j}, D^m_{12j}, D^m_{21j}, D^m_{22j}]$, let us define the relation $R_k = R_i \; x \; R_j$ obtained by *intersection* as follows:

$$R_k = [RQ_k, D^m_{11k}, D^m_{12k}, D^m_{21k}, D^m_{22k})]$$

where RQ_k is a relation belonging to SAc obtained by intersecting RQ_i and RQ_j. It has to be noted that a SAc relation is given by a set of atomic relations of the interval algebra: then $RQ_i \cap RQ_j$ is the intersection of the two sets. As the sub-algebra SAc is closed under the operation of intersection, $RQ_k \in$ SAc.

D^m_{11k} is equal to $D^m_{11i} \cap D^m_{11j}$ and similar relations hold for the other distances. The intersection is empty if one of the four distances is empty and/or RQ_k is the empty relation.

Definition. A binary constraint is in *canonical form*, if it is the disjunction of IDSA relations with intersections mutually empties, i.e., if the intersection of any two relations R_i and R_j composing the constraint, is empty.

All IDSA relations in a binary constraint satisfy this important property: they are pairwise not-unitable (i.e. they are in their *canonical form*). If two IDSA relations can be joined together, as they are processed, they are packed into a single relation. As previously explained, a relation of IDSA is a tuple of four distances between the extremes of two intervals:

$$R_i = (D_{11}, D_{12}, D_{21}, D_{22})$$

where $D_{ij} = d_{ij}(L_{ij}, U_{ij})$, for $i, j = 1, 2$.

To identify an IDSA relation, the tuple of distances must be consistent; to check the consistency of a single IDSA relation, the system follows these steps:

1. the four distances composing the tuple are represented over a network whose nodes are points (the extremes of the intervals) and whose arcs are labelled by the distances specified in the tuple, thus obtaining a Distance Algebra network. If this network is consistent, the tuple is consistent, too, and it represents an IDSA relation between two intervals;

2. a minimal network algorithm (as described in the next section) is then applied to this point-based network in order to compute the tightest constraints over the point-network: a new tuple is obtained which is equivalent to the initial one (i.e. represents the same set of solutions for the point-network).

This corresponds to determine the canonical form of a binary relation: let us say, the minimal tuple. Indeed, a relation in canonical form is a particular element of a class of equivalence which contains all the tuples that can be represented by a four-point network with the same set of solutions.

Thus, an IDSA relation can be viewed as a class of equivalence, rather than as a single tuple of distances.

The universal relation Φ is given by:

$$\begin{aligned} \Phi \;=\; & (d(minf, pinf), d(minf, pinf), \\ & d(minf, pinf), d(minf, pinf)) \end{aligned}$$

The empty relation, ϵ, is represented by the set of all the inconsistent tuples. A tuple is inconsistent when it implies a null or negative duration for the constrained intervals. For example, the tuple:

$$(d_{11}(5, 10), d_{12}(3, 4), d_{21}(minf, pinf), d_{22}(minf, pinf))$$

is inconsistent.

The identity relation I is:

$$I = (d_{11}(0, 0), d_{12}(o(0), pinf), d_{21}(minf, o(0)), d_{22}(0, 0))$$

where $o(0)$ means that the extreme is open.

In general, the union of two IDSA relation is not an IDSA relation. Hence, IDSA is a sub-algebra, not an algebra, because it is closed under operations of intersection and composition, contains the empty relation, the universal relation, and the identity relation, but it is not closed under union.

In an IDSA network, each edge is labelled by a disjunction of one or more IDSA relations. This set of relations is maintained in a form such that the relations are mutually not-unitable (two IDSA relations are unitable if it is possible to express them as a single IDSA relation). In this way the information is maintained as compact as possible, because each edge is labelled by a number of relations which is as small as possible.

Let us analyse the relantionships of the IDSA sub-algebra with both the interval algebra IA and the SAc sub-algebra. First of all, it should be noted that a class of equivalence of tuples (i.e., an IDSA relation) identifies one and only one relation belonging to the sub-algebra SAc: in fact, each distance determines a PAc relation between the two constrained points. Thus, an IDSA relation between two intervals determines a PAc network over four nodes (the extremes of the intervals). Let us call it P_n. Between two intervals surely holds one of the 2^{13} qualitative relations of IA; this relation is represented by a conjunction of relations of PAc over the endpoints of the intervals by the network P_n: this means that the qualitative relation we are looking for (i.e. the one represented by the tuple of distances) is a relation belonging to SAc. Thus, an IDSA network is as expressive as an IA network, if its use is limited to qualitative relations.

Comparing IDSA with DA, we can say that, if each point is viewed as an extreme of an interval, all relations that can be expressed within a TCSP can also be expressed within an IDSA network, while the vice-versa is not true, because in a TCSP metric relations

between intervals cannot be represented.
We can then conclude that an IDSA network (in which constraints are disjunction of IDSA relations) has a greater expressive power than both IA and DA representations.

4 Algorithms

In this section, we illustrate the algorithms used for the various operations over the netwok and their complexity. Different queries can be asked to the temporal reasoner: to check consistency, to find a possible solution, to compute the minimal network.
A temporal network is consistent if there is a consistent labelling, i.e. if at least one singleton labelling exists which satisfies all the constraints. A singleton labelling is obtained by considering only one relation for each binary constraint and only one duration for each interval. It is consistent if there is a possible assignment of the position of the intervals onto the time line, such that all the constraints forming the labelling are satisfied. For each singleton labelling, the network can be translated into a network over time points (extremes of intervals), i.e. an STP (Simple Temporal Constraint Problem) [9]. The network is consistent if and only if a consistent STP exists. Checking consistency requires an algorithm complexity $O(n^3)$, where n is the number of nodes in the STP (path-consistency). In our system, consistency of STP is checked using the arithmetic constraint solver offered by the CLP(R) language [10], the language utilized to implement the temporal reasoner (as well as the planner). Consistency is checked in an incremental way. Often it is unnecessary to determine a complete labelling of the network to discover its inconsistency: as soon as inconsistency is detected, that labelling is ignored, and a new one taken into consideration. Thus, consistency check becomes a simple search with backtracking.
The problem of finding a solution for a given network can be dealt with in different ways. One can consider a "solution" as a consistent singleton labelling, for example. But even within a singleton labelling there is some uncertain information: if we say that an interval can assume only one possible duration selected from a disjunction of durations, we must remember that this duration itself is affected by uncertainty, because it is expressed as d(L, U), which means that the interval duration can vary from L to U. Then, as a possible solution of a network, we intend the set of exact values that the extremes of intervals can assume.
We refer to a given time line, which is defined as a particular interval called I_0, such that all intervals of the network are contained within I_0. If we assign the value zero to the starting point of I_0, when a singleton labelling is fixed, a range of variability for each extreme of each interval can be found, and a precise value belonging to this range can be imposed for each point. Thus, finding a solution of a given network means to assign to each extreme of each interval a precise value within the time line, in such a way that all temporal constraints are satisfied.
The minimal network is computed according to:

1. search for all consistent singleton labellings;
2. for each singleton labelling:
   (a) translate it into an STP, whose nodes represent the interval extremes and whose arcs are labelled with the range of variability of the distances between nodes.
   (b) apply path-consistency over the STP to obtain its minimal network;
   (c) translate the time points network into the equivalent interval network
3. the minimal network is obtained as the union of all these networks.

The problem of checking the consistency of an IDSA network is a NP-hard problem: in the worst case it is necessary to check the consistency of all singleton labellings, to conclude that the network is inconsistent; if e is the number of edges on the graph, and k is the maximum number of IDSA relations per arc, k^e is the maximum number of singleton labellings to be checked. For each singleton labelling, a path-consistency algorithm is applied, so that the complexity of the whole problem, in the worst case, is $O(n^3k^e)$. Obviously, the lower the number of relations per arc, the lower the number of singleton labellings k^e. Thus, we can understand the reason why it is important to maintain the knowledge as "compact" as possible, that is to represent a binary constraint by means of a small number of disjunctive relations, as previously mentioned.
The algorithm complexity is still $O(n^3k^e)$ in the case of finding a solution and in that of computing the minimal network. In the case of the minimal network, even if the time complexity is equivalent to the ones defined for the worst case of the problems of checking consistency and finding a solution, it is worth noting that, all singleton labellings must be always examined, so that the time needed to find the minimal network is always greater than the time needed to check consistency or finding a solution.

5 Other Approaches

Other approaches to temporal representation and reasoning allow us to represent explicitly metric information and to combine it with the qualitative one.
In [11], the information is represented in two separate networks: the first with qualitative information, whose nodes represent intervals, and the second with metric information, whose nodes represent ending and starting points of intervals. Within this framework it is possible to represent single time points, that is time points which not necessarily are the extreme of an interval. This is not allowed in our representation: in a IDSA network only intervals with a duration greater than zero can be represented.
Except this, our representation has a greater expressive power than Kautz and Ladkin's one. The sublanguage L_a (only qualitative) can be represented in

an IDSA network, because all IA relations can be expressed in our formalism as we have shown. Moreover, simple metric constraints between endpoints can surely be represented in an IDSA network, because distances between endpoints can be specified within an IDSA relation. The vice-versa is not true: an IDSA network cannot be translated into Kautz and Ladkin's formalism because in their formalism a disjunction of metric relations between intervals cannot be represented, neither a disjunction of metric relations between a pair of points can be represented. Only one singleton labelling of an IDSA network can be expressed within this formalism, because it is equivalent to an STP (thus, only the sublanguage L_m is sufficient, in this case).

In [12], only one network with two kinds of nodes is used: one representing intervals, the other temporal points. This approach allows the direct representation of qualitative and metric information, but it forces us to consider three nodes for each interval in the network: one representing the interval, and two representing its extremes. Our representation can be considered hybrid in a different way: at the inner level, where propagation algorithms are applied, only metric constraints are represented, in the form of distances between ending points of intervals. However, at the interface level, both in input and output, this information is split into the qualitative and metric forms. This is because it may be useful for the user, in input, to represent only the partial knowledge about temporal relations at hand (e.g. that something is before something else) and, in output, to easily interpret and use the qualitative constraints between intervals (instead of points, assumed as primitives, e.g. in [8]).

As for the complexity, in our framework the number of singleton labellings to be examined is always less than or equal to the number of singleton labellings of a Meiri's network: indeed, each IDSA relation (the disjunction of IDSA relations forms a constraint on an IDSA network) may represent the union of two or more relations that are expressed as a disjunction on a Meiri network.

As an example, let us show how the typical temporal reasoning problem proposed in [12] can be dealt with in our representation. The problem is:

John and Fred work for a company in LA. They usually work at the local office, in which case it takes John less than 20 minutes and Fred between 15-20 minutes to get to work. Twice a week John works at the main office, in which case he commutes at least 60 minutes to work. Today John left home between 7:05-7:10, and Fred arrived at work between 7:50-7:55. We also know that Fred and John met at a traffic light on their way to work.

Taking into account that the beginning of the story is at 7:00 a.m., let us consider a reference interval I_0 of undefined duration starting at the beginning of the story. The input knowledge concerning intervals (during which temporally qualified facts take place), can be translated as follows:

intervals

- I_0 $d(0, +\infty)$ - reference interval
- I_1 $[d(0, 20), d(o(60), +\infty)]$ - John is going to work
- I_2 $d(15, 20)$ - Fred is going to work

relations

- $I_1[d, d_{11}(5, 10), d_{12}(-\infty, +\infty), d_{21}(-\infty, +\infty), d_{22}(-\infty, +\infty)]I_0$ - John left home between 7:05-7:10
- $I_2[d, d_{11}(-\infty, +\infty), d_{12}(50, 55), d_{21}(-\infty, +\infty), d_{22}(-\infty, +\infty)]I_0$ - Fred arrived between 7:50-7:55
- $I_1[ct, d_{11}(-\infty, +\infty), d_{12}(-\infty, +\infty), d_{21}(-\infty, +\infty), d_{22}(-\infty, +\infty)]I_2$ - I_1 and I_2 have a non empty intersection.

The relation d is the usual qualitative relation *during*, while ct is the qualitative relation *contemporary* representing the set $\{o\ fi\ di\ si\ =\ s\ d\ f\ oi\}$ that can be translated in the following tuple:

$$(d_{11}(-\infty, +\infty), d_{12}(0, +\infty), d_{21}(-\infty, 0), d_{22}(-\infty, +\infty))$$

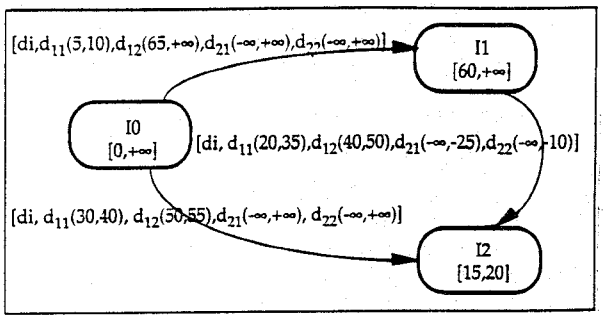

Figure 1. The output temporal information.

Then, on the basis of the obtained results (figure 1), it is possible to derive that:

- the information in this story is consistent,
- John arrived at main office after 8:05 a.m. (relation between I_0 and I_1),
- Fred left home between 7:30-7:40,
- John arrived at work at least 10 minutes after Fred, etc.

A work proposed by Becker (1995) deals with a hybrid formalism to treat both qualitative and quantitative temporal and non-temporal constraints. The aim is to combine Allen's and Rit's formalisms [13] and nets of arbritary constraints. It seems to us that, being events in the Allen's and Rit's approaches quite heterogenous entities from an ontological point of view, their unification in a framework might be problematic. Instead, in our formalism, we extend the Allen temporal logic only by including, let us say, a measure over intervals. We preserve the ontology concerning the temporally qualified facts (i.e. properties, events and processes). Besides, taking into account that a SOPO (events are defined as sets of possible occurrences) consists of 6 components, stating a minimal (-) and a maximal (+) value for start (S), duration (D) and finish (F) [13], and that this information has to be integrated with other of different type (qualitative, non-temporal) [7], the propagation algorithm complexity seems to increase rapidly (but in [7] a detailed analysis of complexity is lacking).

6 Summary and Conclusions

We gave in this article a general frame for dealing with possibly uncertain qualitative and quantitative constraints relative to temporal intervals. We have shown that the our representation, blending together the two type of information in a homogeneous way, has an expressive power greater than both Interval Algebra and Distance Algebra representations. By defining what we called the canonical form of IDSA relations, we kept temporal information as compact as possible, i.e., the number of disjunctions for each constraint as small as possible, so as to reduce computational effort. We verified that this property of the temporal representation yields a better efficiency of the temporal reasoner when used in planning and scheduling problems. In these applications, knowledge is usually uncertain and coarse, and the notion of disjuntion is needed to represent alternatives and/or mutual exclusions.

We applied the temporal reasoner to solve a problem of temporal scheduling of a set of industrial operations in a manufacturing application (as shown in an extended version of this paper submitted to a journal). Dealing with problems (e.g. in the robotic field) which require efficient planning of activities over time in the presence of complex domain-dependent constraints, planning and scheduling technologies have to be combined in order to find not only which activities to execute to achieve a given goal, but also when those activities should be executed. Then, in these applications, we included our temporal reasoner in a more general architecture and we used it as the temporal constraint solver of the planning system [4], [5]. Ordering temporally qualified actions falls out from maintaining the consistency of a temporal constraint network and by computing the minimal network. Thus, the planning system generates a plan taking into account partial or general deadlines. We applied our system to plan the actions of a mobile robot used for maintenance interventions over hydraulic circuits in an experiment carried on within the framework of the Italian Advanced Project in Robotics of the CNR (MANUEL Project) [6].

Acknowledgments

This work has been developed at LADSEB-CNR of Padova. It has been partially funded by the Italian Special Research Programs on Robotics and Automated Planning of the CNR, and by the Italian Ministry of University, Science and Technology. We would like to thank Paolo Bison for many helpful advices.

References

[1] J.F. Allen, "Maintaining Knowledge about Temporal Intervals", *Comm. of the ACM*, Vol. 11, 26, pp. 832-843,1983.

[2] J.F. Allen, "Towards a general theory of action and time. *Artificial Intelligence*, Vol. 23, pp. 123-154,1984.

[3] J.F. Allen, H.A. Kautz, R.N. Pelavin, J.D. Tenenberg, *Reasoning about Plans*, Morgan Kaufmann Publishers Inc., San Mateo, CA, 1991.

[4] S. Badaloni, E. Pagello, L. Stocchiero, A. Zanardi, "Planning temporally qualified robot actions", *Proc. Int. Conf. on Advanced Robotics ICAR93*, Tokyo, Japan, 1993.

[5] S. Badaloni, M. Berati, "Dealing with time granularity in a Temporal Planning System", *Lecture Notes in Artificial Intelligence*, Vol. 827, pp. 101-116, Springer Verlag, 1994.

[6] S. Badaloni, M. Berati, P. Bison, E. Pagello, "Templar: a temporal planner for monitoring the actions of an autonomous robot used in plant experiments", *Proc. 4th Int. Conf. on Intelligent Autonomous Systems IAS-4*, Karlsruhe, Germany, 1995.

[7] S. Becker, "Hybrid temporal and non-temporal knowledge for scheduling" *Proc. of the TIME-95*, Melbourne, FL, 1995.

[8] V. Brusoni, L. Console, B. Pernici, and P. Terenziani, "LaTeR: a general purpose manager of temporal information", *In Methodologies for Intelligent Systems*, Vol.8, LNCS 869, Springer Verlag, 1994.

[9] R. Dechter, I. Meiri, J. Pearl, "Temporal Constraints Networks", *Artificial Intelligence*, Vol. 49, pp. 61-95, 1991.

[10] J. Jaffar, M.J. Maher, P. Stuckey, R. Yap, "Projecting CLP(R) constraints", *IBM Research Report*, 1992.

[11] H.A. Kautz, P.B. Ladkin, "Integrating Metric and Qualitative Temporal Reasoning", *Working Notes AAAI Spring Simposium Series*, pp. 67-78, 1991.

[12] I. Meiri, "Combining Qualitative and Quantitative Constraints in Temporal Reasoning" *Proc AAAI-91*, Anaheim, CA, pp. 260-267, 1991.

[13] J.F. Rit, "Propagating temporal constraints for scheduling", *Proc. of the AAAI-86*, Philadelphia, pp. 383-388, 1986.

[14] A. Trudel, "A formal specification of Allens processes", *Proc. Second Symposium on Logical Formalization of Commonsense Reasoning*, Austin, Texas, 1993.

[15] P. vanBeek, R. Cohen, "Exact and approximate reasoning about temporal relations", *Comput. Intell.*, Vol. 6, pp. 132-144, 1990.

[16] P. vanBeek, "Reasoning about qualitative temporal information", *Artificial Intelligence*, Vol. 58, pp. 297-326, 1992.

[17] M. Vilain, H. Kautz, "Constraint propagation algorithms for temporal reasoning", *Proc. AAAI-86*, Philadelphia, PA, pp. 377-382, 1986.