AN INSTANCE OF THE CLP(X) SCHEME WHICH ALLOWS TO DEAL WITH TEMPORAL REASONING PROBLEMS

Francisco Ibáñez, Raymundo Forradellas, Luis Rueda

Universidad Nacional de San Juan, Argentina.
e-mail: {fibanez,kike,lrueda}@iinfo.unsj.edu.ar

ABSTRACT

In many applications of temporal reasoning is necessary to express metric and symbolic temporal constraints among temporal objects whether they are points or intervals. In order to cope with these requirements different formalisms have been issued, those that allow to express symbolic temporal constraints by one hand, and others involving metric temporal constraints. Although this formalism are suitable to represent just some kind of problems, in many cases, it is necessary to handle and represent in the same framework both metric and symbolic constraints among temporal objects, whether they are point or interval.

Starting from the previous schemes, different formalisms to integrate metric and symbolic temporal constraints have been issued. A common limitation of these proposals is that none of them allows to represent disjunctive constraints involving a metric component and a symbolic one. This type of constraints arises for example in scheduling problems, where an activity must be performed before or after another activity, but considering the setting time of the used resources [Ibáñez,92b].

Besides in many planning applications, the formulation of the problem itself, must be expressed as logic formulas with a period of time associated. Therefore, a temporal reasoning system oriented to planning should be able to express both the logic and the temporal part in a same frame. Unfortunately, none of the approaches to integrate symbolic and metric temporal constraints allows to express the logic part of the problem.

The main aim of this paper is to define a temporal tool which allows to express and unify metric and symbolic temporal constraints among temporal objects (intervals and points). The temporal model proposed in this paper is based on intervals. However, as opposed to other formalisms, the duration of the intervals may be zero, and therefore temporal points are included. In other words, the concept of temporal interval used in the literature (where the duration is strictly greater than zero), is generalized.

Starting from the temporal model, a new operational framework oriented to the resolution of the problems rather than focused to the representation of temporal reasoning problems is defined. The proposed operational framework was designed as a new instance of the CLP(X) scheme [Ibáñez,93] in which the computational domain is formed from temporal objects. Conceptually, the variables of the CLP(Temp) language have associated a finite set of pairs of values representing temporal intervals.

INTRODUCTION

The goal of this paper is to define a temporal tool that allows to express and to reason on symbolic and metric temporal constraints between points and intervals of time.

There are two main subgoals:

- A new temporal model.
- An operational model which is based on the temporal model defined previously.

The first subgoal consists of the definition of a new temporal model in which points and temporal intervals as well as symbolic and metric temporal constraints can be expressed in a unified way.

Among different approaches providing an suitable operational frame, the CLP(X) scheme is chosen. That is, starting from the temporal model we define a new instance of CLP(X) scheme called CLP(Temp), where *Temp* denotes the temporal domain. This operational framework allows to express the reasoning part of the constraints of the proposed temporal model.

The applicability of the proposed temporal model and the respective operational model is focused on the planning area, which normally involves complex temporal constraints.

In the domains of temporal reasoning problems, such as planning, it is necessary to represent and to reason on the dynamic part of the problem. This requires a temporal object (or primitive) to be described and to be expressed and reasoned on the temporal relations. Therefore different algebras of temporal relations were defined, such as the interval algebra [Allen,83a][Ladkin,87], the point algebra [Vilain,86], and logic theories based on these representations, specially the time and action theory [Allen,84].

The point algebra as well as the interval algebra, use one constraint network where the nodes are associated to the temporal objects (points or intervals respectively), and the edges are associated to the symbolic constraints among these objects. The operational model of these algebras are based on the elimination of inconsistencies by using symbolic relation transitivity tables.

However, in several planning and scheduling problems distances among temporal objects need to be expressed. In other words, the definition of a *metric* on the time has to be involved. The temporal logic defined in [MacDe,82] takes into account a temporal line, with its correspondent metric, in which the events are located and the relative temporal distances among them could be defined. The application of linear programming [Valdés,86] and specific methods of resolution of disequations [Malik,83] for the resolution of temporal problems involving distances allowed to provide the McDermott theory a practical framework and later on originated a temporal manager called Time Map Manager (TMM) [Dean,87] widely used in several areas of AI. An extension of this representation scheme was realized on the model presented in [Decht,91], which allows to express and manage disjunctions on relative distances among successes.

Starting from the previous distinction, two different schemes can be considered in order to represent temporal constraints: symbolic or qualitative representation schemes and metric or quantitative representation constraints. Although the mentioned formalisms are adequate for representing some restricted types of problems, in very important areas such as planning and scheduling, it is necessary to represent in a same application, points and temporal intervals as well as metric and symbolic constraints among them.

For example, in planning and scheduling problems, it is necessary to represent that an activity (which has got associated a temporal interval) is realized before or after other that uses the same resource, besides the latter must finish a certain time before the end of the project. The first one is a disjunctive symbolic constraint among intervals, while the second one is a metric constraint between a time interval (which represents the activity in the project) and a time point (which represents the instant of the end of the project). In addition, the metric information is needed in order to restrict the time in which an activity can start or the maximum or minimum time between the performing of two activities. Therefore, it arises the necessity of expressing symbolic and metric constraints in a same formalism, as well as temporal interval and points.

Up to now, four basic formalism have proposed the integration of symbolic and metric temporal constraints [Ladkin,89], [Kautz,91], [Meiri,91] and [Tolba,93]. The main advantage of these approaches is that they provide not only a representational but also an operational framework.

These proposes just allow to represent metric and symbolic constraints. A common limitation is that they do not allow to represent metric-symbolic constraint among temporal intervals, that is, constraint involving a symbolic part and a metric part. This type of constraints arises, for example, in planning and scheduling problems in which activities must be disjunctive, but taking into account the setting time of the resources. In other words, an operation can be performed before or after other one, but with a minimum time between the end of an activity and the start of the following, which represents the setting time of the resource.

Besides in several planning applications the formulation of the problem is expressed by logic formulas associated to a period of time, and therefore an oriented-planning temporal reasoning system should be able to express in a same framework both the logic and the temporal part.

This integration of the logic and the temporal part of a temporal reasoning problem, can not be expressed in the framework of the previous formalisms.

Starting from the previous requirements, it arises the necessity of expressing in a same framework, the logic and the temporal part related to an oriented-planning temporal reasoning system.

The expressiveness required to the temporal component must include the unification of temporal interval and points as well as the integration of symbolic and metric constraints. The latter refers to the possibility of expressing both metric constraints among temporal objects and symbolic constraints among them, as well as the capability of expressing symbolic-metric constraints among temporal objects.

The required tool is divided in two parts:

1. The definition of a new temporal model that unify temporal interval and points, and integrate symbolic and metric temporal constraints among these temporal primitives.

2. The definition of an operational model based on the temporal model defined previously, which is oriented not only to the representation of temporal reasoning problems, but also to the resolution of them.

1 THE TEMPORAL MODEL

In this section we present an approach that first unifies points and temporal intervals and also symbolic and metric temporal constraints.
The temporal model defined here is finite, discrete and not lineal [Ibáñez,92a] [Ibáñez,93].
A new class of constraints between temporal intervals is presented. Starting from this new class of temporal constraints we define a temporal model which is formalized as a structure composed by a domain based in temporal intervals and the interpretation of temporal relations.

1.1 DEFINITION OF A TEMPORAL STRUCTURE

Definition 1

We define a *temporal interval* as a pair $<s,d>$, where $s,d \in N^0$.
N^0 represents the set of natural numbers including zero.
The components *s* and *d* represent the start and the duration of the temporal interval $<s,d>$, respectively.

Definition 2

We define the set of all possible temporal intervals as the set:

$$I = \{<s,d> : s,d \in N^0 \}$$

A variable which take its values in the set I, will be called either: *temporal-interval variable* or, simply, *interval variable*.

Definition 3

Let $<s,d>, <s',d'> \in I$ be two temporal intervals, and let $X,Y \subseteq N^0$ be two sets, we define the following relation symbols:

BEF(*X*), OVERL(*X*), DUR(*X*, *Y*), DUR_I(*X*, *Y*), OVERL_I(*X*) y AFT(*X*)[1],

with arity two, as follows:

$<s,d>$ **BEF(X)** $<s',d'>$	iif	$\exists x \in X: s+d+x = s'$
$<s,d>$ **OVERL(X)** $<s',d'>$	iif	$\exists x \in X: s'+x=s+d \wedge s<s' \wedge s+d<s'+d'$
$<s,d>$ **DUR(X,Y)** $<s',d'>$	iif	$\exists x \in X: s'+x=s \wedge s+d+y=s'+d'$

And its inverses:

$<s,d>$ **AFT(X)** $<s',d'>$	iif	$\exists x \in X: s'+d'+x=s$
$<s,d>$ **OVERL_I(X)** $<s',d'>$	iif	$\exists x \in X: s+x=s'+d' \wedge s'<s \wedge s'+d'<s+d$
$<s,d>$ **DUR_I(X,Y)** $<s',d'>$	iif	$\exists x \in X: s+x=s' \wedge s'+d'+y=s+d$

BEF, OVERL, DUR and their inverses, express the symbolic information between related intervals, while the arguments X e Y, referred as characteristic distances, express the quantitative information.

[illegible] distances are formed by unary sets, the quantitative information determines an exact relative position between the intervals, as it is shown in figure 1.
Examples:

$<2,5>$ BEF({3}) $<10,20>$ is true.

$<2,9>$ OVERL({3}) $<10,20>$ is false.

$<2,5>$ BEF({2,3,4,5}) $<10,20>$ is true.

BEF(X), OVERL(X), DUR(X,Y) and its inverses, will be referred as not disjunctive symbolic-metric relation symbols. As an abuse of notation, in occasions we will simply refer to the relation symbol as relations.

---

[1] BEF, OVERL, DUR, DUR_I, OVERL_I, y AFT, mean the relations 'before', 'overlaps', 'during', 'during inverse', 'overlaps inverse' y 'after', respectively.

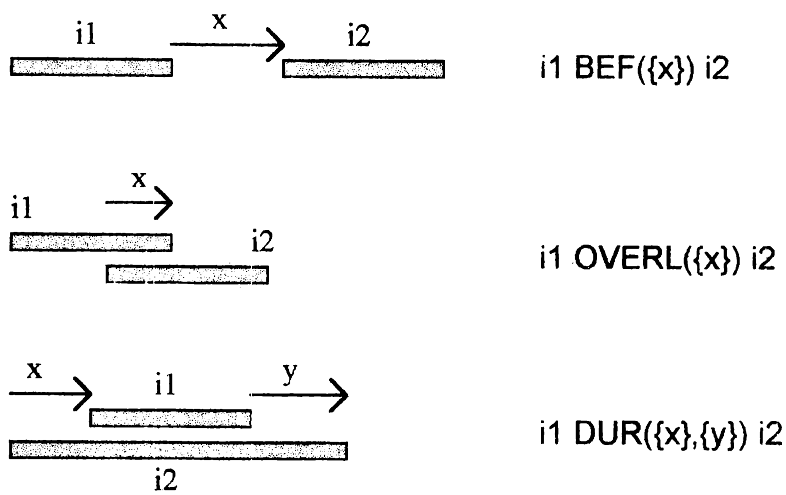

Figure 1. Characteristic distances formed by unary sets

It should be noted that the arguments express the metric part of the relation.
Not disjunctive symbolic-metric relations as BEF(N^0), OVERL(N^0), DUR(N^0,N^0), DUR_I(N^0,N^0),OVERL_I(N^0) and AFT(N^0), do not specify any metric constraint and they will be also referred as *not disjunctive symbolic* relations denoted as BEF, OVERL, DUR, DUR_I, OVERL_I y AFT[2].

Allen Relation	**Symbolic-metric Relation**
'before'	BEF(N)
'meets'	BEF({0})
'during'	DUR(N,N)
'start'	DUR({0},N)
'finish'	DUR(N,{0})
'overlaps'	OV(N)
'equal'	DUR({0},{0})

N denotes the set of natural numbers without zero

Table 1 : Allen relations expressed as particular cases of symbolic-metric relations.

All Allen relations can be expressed as particular cases of the relations defined previously, as shown in table 1. The inverse relations are defined in a similar way.
The following definition express the disjunctive relations between temporal intervals.

Definition 4
Consider <s,d>, <s',d'> $\in$ I, and let $r_1 , \ldots , r_k$ ($1 \leq k \leq 6$) be not disjunctive symbolic-metric relations symbols with different symbolic component. The relation symbol $\{r_1 , \quad , r_k\}$ is interpreted as follows:

<s,d> $\{ r_1 , \ldots , r_k \}$ <s',d'> iif <s,d> r_1 <s',d'> v ... v <s,d> r_k <s',d'>

---

[2]This notation should not be confused with what is normally user to represent the Allen relations.

The relation symbol denoted as $\{ r_1 , \ldots , r_k \}$ is called *general symbolic-metric* relation symbol.

For example:

{ BEF({3}), DUR({3}, {3,7,8,9}), AFT({7,8,9}) }

is a general symbolic-metric relation symbol, and

$\langle 2,5\rangle$ { BEF({3}), DUR({3}, {3,7,8,9}), AFT({7,8,9}) } $\langle 10,20\rangle$

is true.

In this section, the symbolic relations have been considered to name the relations. A general symbolic-metric relation is disjunctive if $k>1$, otherwise it includes just one symbolic relation (if $k=1$).

Definition 5

Let $\langle s,d\rangle \in I$ be an interval. In order to include unary relations, we define the following relation symbol:

$$\{ \in \mathrm{Set}(\mathrm{Start}, \mathrm{Dur}, \mathrm{Max}) \}_{\mathrm{Start},\mathrm{Dur} \subseteq N^0,\ \mathrm{Max} \in N^0}$$

interpreted as follows:

$$\langle s,d\rangle \in \mathrm{Set}(\mathrm{Start}, \mathrm{Dur},\mathrm{Max}) \text{ iif } s\in \mathrm{Start} \wedge d\in \mathrm{Dur} \wedge s+d\leq \mathrm{Max}$$

Intuitively, Set(Start, Dur, Max) is the set of temporal intervals such that their possible starts belong to the set *Start*, their possible duration belong to the set *Dur* and their possible ends are not greater than *Max*.

Examples:

$\langle 3,6\rangle \in$ Set({1,2,3,4},{4,5,6},9) is true.

$\langle 3,6\rangle \in$ Set({1,2,3,4},{4,5,6},8) is false.

The relation symbols introduced in definitions 4 and 5 constitute a *signature* denoted as Σ_{Temp}. The domain I (def. 2) with the symbol interpretations which constitute the signature define the structure denoted as INT.

For example, under the structure INT,

$\langle 2,5\rangle$ { BEF({3}), DUR({3}, {3,7,8,9}), AFT({7,8,9}) } $\langle 10,20\rangle$

is true and denoted as:

$\models_{INT}$ $\langle 2,5\rangle$ { BEF({3}), DUR({3}, {3,7,8,9}), AFT({7,8,9}) } $\langle 10,20\rangle$

1.2 DEFINITION OF THE TEMPORAL MODEL

The proposed temporal model is formed by:

- Variables, taking their values in the set I.
- Constraints, built out of variables and the signature Σ_{Temp}.

Formally, constraints can have the form:

I_i $\{r_1, ..., r_k\}$ I_j or
$I_i \in$ Set (Start, Dur, Max)

where I_i, I_j are interval variables.

The assignation to variables $\{I_i \leftarrow <s_i,d_i>, I_j \leftarrow <s_j,d_j>\}$ satisfies the constraint $I_i \{r_1, ..., r_k\} I_j$ if $<s_i,d_i> \{r_1, ..., r_k\} <s_j,d_j>$ is true under the structure INT.
Similarly, the assignation $\{I_i \leftarrow <s_i,d_i>\}$ satisfy the constraint $I_i \in$Set(Start,Dur,Max) iif $<s_i,d_i> \in$ Set(Start,Dur,Max) is true under the structure INT.

Finally, given a set of constraints involving the variables $I_1, ... , I_n$, we say that this set of constraint is INT-satisfying if exists an assignation $\Phi=\{I_1 \leftarrow <s_1,d_1>, ..., I_n \leftarrow <s_n,d_n>\}$ which satisfy all the constraints. Φ is called INT-satisfier.

Deliberately, the temporal model presented in this section does not include any axiom. The axioms are related to the operational model [Ibanez,94] and it is not included in this paper.

Example:
Consider a very simple example of scheduling with just three operations, denoted op1, op2 and op3 with duration 40, 60 and 50 t.u.. All the operations must be performed in an occupation space (0,100). That is, the minimum starting time is 0 t.u. and the maximum finishing time of each operation is 100 t.u..

The constraints between operations are the following:

1. The operation op1 and the operation op2 must be disjunctive or they can overlap at least 3 t.u.
2. The operation op1 must finish 10 t.u. before the operation op3 or it can start 10 t.u. after op3 is finished.
3. The operation op3 must start while op2 is being performed and it must finish after op2 is performed.

The previous problem can be easily expressed by the proposed temporal model as follows:
The duration and occupation space define the unary constraints while the constraints between the operations define the binary constraints.
Unary constraints:

$I_1 \in$Set({0,...,60},{40},100)
$I_2 \in$Set({0,...,40},{60},100)
$I_3 \in$Set({0,...,50},{50},100)

Binary constraints:

I_1 {BEF,OVERL({1,2,3}),OVERL_I({1,2,3}), AFT} I_2 (1)
I_1 {BEF({10}), AFT({10})} I_3 (2)
I_2 {OVERL} I_3 (3)

Domains defined by unary constraints for the variables I_1, I_2 ,I_3 are respectively:

{<0,40>, <1,40>, ..., <60,40>},
{<0,60>, <1,60>, ..., <40,60>}, and
{<0,50>, <1,50>, ..., <50,50>}

The variable assignation {I1 $\leftarrow$ <0,40>, I2 $\leftarrow$ <37,60>, I3 $\leftarrow$ <50,50>} verifies all the constraints and therefore it is an INT-satisfier of the set of constraints.
In the context of scheduling, this variable assignation constitutes a possible solution. The figure 6.2 shows graphically this solution.

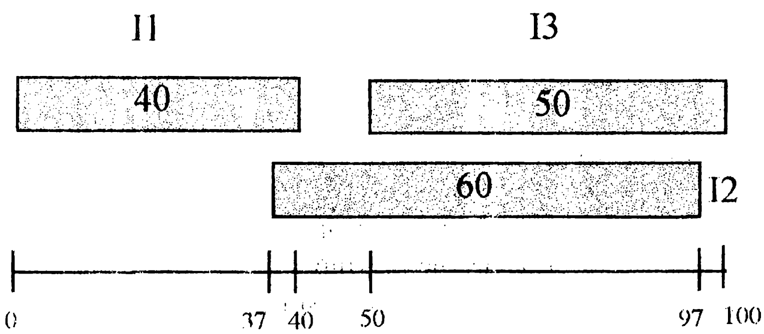

Figure 6.2. A factible solution

Note that this example involves symbolic-metric constraints between intervals, and therefore *they can not be expressed in other approaches* [Meiri,91], [Kautz,91], [Tolba,93]
The *previous* example is quite easy, but it shows the simplicity and expressiveness of the temporal model.

1.3 EXPRESSIVENESS OF TEMPORAL MODEL.

1.3.1 UNIFING POINTS AND INTERVALS.

The proposed temporal model allows to define, in an unified way, points and intervals. A temporal point is simply a particular case of a temporal interval where the duration is zero. For example, the unary constraint $P1 \in Set(\{a_1,..., a_n\},\{0\}, a_n)$, defines a temporal point P1 which is undetermined between a_1 and a_n t.u. Formally, P1 is an interval variable defined as a duration equal zero. The domain which is associated to this variable is $\{<a_1,0>,....,<a_n,0>\}$.
Since the unary constraints can involve duration equal zero, it is not necessary to distinguish between points and intervals. That is, an interval variable, initially declared with a duration including zero, can be converted to the representation of a temporal point if the duration is reduced to zero.
It is important to note that none of the existing proposals allows this possibility.
By the other hand, in order to represent a success or property involving a duration greater than zero, it is only necessary to state an interval variable with a constraint excluding which exclude the value zero.
The capability of unifying points and intervals allows to unify not only temporal objects but also constraints.
Unlike Meiri's proposal, it is not necessary to distinguish symbolic and metric relations between point, between interval and those that involve a point and an interval. In our approach there exists a unique class of constraints.
The following sections are dedicated to each of distinct types of constraints and they show how they are represented in the proposed temporal model.

1.3.2.1 Symbolic constraints

1.3.2.1.1 Symbolic constraints between points

Symbolic relations between points are formed starting from the not empty subsets of the set {<,=,>} and they can be expressed as it is shown in table 2.

P1{<}P2	P1{BEF(N)}P2
P1{<,=}P2	P1{BEF}P2[3]
P1{<,>}P2	P1{BEF(N), AFT(N)}P2
P1{<,=,>}P2	P1{BEF,AFT(N)}P2
P1{=}P2	P1{BEF(0)}P2
P1{>}P2	P1{AFT(N)}P2
P1{=,>}P2	P1{AFT}P2

Table 2. Representation of symbolic relation between temporal points.

Although in the proposed formalism these constraints can be represented directly by a unique constraint between the points which it connects, the representation used by Meiri is more direct.

1.3.2.1.2 Symbolic constraints between intervals.

As we saw previously (table 1), all Allen relations are particular cases of general symbolic-metric relations. However, since six basic relations are only necessary in our approach, the representation is more compact. For example, to represent that "John and Fred meet during a moment in a bar", [Meiri,91] use the relation:

{start, start-i, during, during-i, finish, finish-i, overlaps, overlaps-i, equal}

In the proposed focus, this relation is expressed as follows:

{OVERL, DUR, DUR-I, OVERL-I}

1.3.2.1.3 Symbolic constraints between points and intervals.

The constraints between a point and an interval are only comparable with the Meiri's approach, since that is the only view that allows to express temporal points as a proper temporal entity. The same considerations and justifications done to constraints between intervals carry out for this case. That is, in both approaches, the symbolic relations between points and interval are directly expressed by mean of just one constraint. However, it is necessary to note that in our approach the representation is more compact because it uses less basic relations.

1.3.2.2 Metric constraints.

1.3.2.2.1 Metric constraints between points.

Unary constraints between points are directly represented by declaring interval variables with duration zero. It is only necessary to use a node and a unary constraint. In the Meiri's approach (the unique proposal that allows to represent temporal points) the constraints must be indirectly expressed by two nodes and a binary constraint between them. Metric constraints between points are directly expressed by a binary constraint between the points that it connects. Metric constraints between points restrict the possible temporal distances between the points that it connects. Those positive distances can be expressed by the relation BEF, while those negative distances can be expressed by the relation AFT.

For example, the constraint:

$$(-12 \leq P2-P1 \leq -10) \vee (-6 \leq P2-P1 \leq -5) \vee (7 \leq P2-P1 \leq 8)$$

is expressed as:

P1{BEF({7,8}), AFT({5,6,10,11,12})}P2.

---

[3]As shown previously, this representation is equivalent to P1{BEF(N^0)}P2, where N^0 represents the natural numbers including zero.

However, there could be more than one possible representation for constraints involving points. This aspect doesn't affect the expressive power of the proposed model, but it must be taken into account in the implementation to improve the efficiency.

1.3.2.2.2 Metric constraints between intervals.

Metric constraints between intervals can be directly expressed by the constraint $I_i \in$ Set($Start_i$,Dur_i,Max_i). Note that in [Ladkin,92], [Kautz,91] and [Meiri,91], the unary constraints between intervals must be expressed not only involving the extremes of them but also considering an additional point as origin. For example, the constraint $I_1 \in$ Set({0,...,90},{10,11,12},100) needs four nodes to be represented, with several unary constraints instead of a unique node with a unique unary constraint.

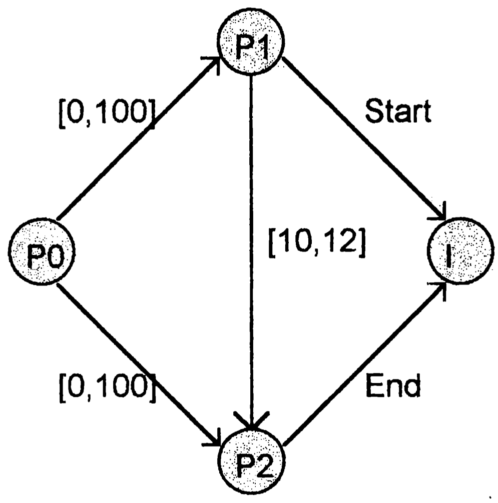

Figure 3. Meiri graph for the constraint i(I1, 0, 100, {10,11,12}).

Figure 3 shows the Meiri graph needed for representing the previous constraint. In the case of Ladkin and Kautz the representation is even more complicated, since points and intervals are maintained in separated networks. It uses 4 nodes and 5 constraints to represent the unary constraint, besides the quantity of constraints respect of the rest of the graph grows considerably and the representation is therefore much more complex.

1.3.2.3 Symbolic-metric Constraints

1.3.2.3.1 Symbolic-metric Constraints between intervals

No disjunctive symbolic-metric constraints between two intervals are directly expressed by only one binary constraint. For example, the constraint I{BEF({10})}J express that the event represented by I must finish 10 t.u. before the event represented by J start.
In the previous approaches it is necessary to use the nodes associated to the extremes of the intervals and to represent the constraint between the intervals, indirectly through these nodes.
Disjunctive symbolic-metric constraint are directly expressed in the our approach. This type of constraints cannot be expressed in none of the previous focus.

1.3.2.3.2 Symbolic-metric constraints between point and intervals

Starting from the fact that in the previous defined temporal model, the temporal points are simply particular cases of temporal intervals in which no disjunctive symbolic-metric constraints between point and interval are directly expressed by only one constraint between the temporal point and the temporal interval. For example, given the constraints for a temporal point P and a temporal interval I, the constraint P {BEF({10})} I express that the event represented by P must be performed 10 t.u. before the event represented by I start. However, in the Meiri's approach the quantity of required temporal objects is greater, since it needs additional points to represent the extremes of the interval.

2 THE OPERATIONAL MODEL

In this section we present a proposal in order to integrate the domain of temporal interval in the framework of Constraint Logic Programming [Ibáñez,94a].

The language proposed is defined by CH-rules, and the formal properties of the language [Ibáñez,94c] are based on the properties of CH-rules. Conceptually, the CLP(Temp) language maintains a close relation with the CLP(FD) instance.

In the CLP(FD) instance, a domain variable is associated a finite set of values. The constraints produce fails, or contrarily, reduce the domains associated to the variables, and eliminate the inconsistent values. The techniques for eliminating inconsistent values are based on the CSP (Constraint Satisfaction Problems).

Conceptually, in a CLP(Temp) language, the variables are associated to a finite set of pair of values representing temporal intervals. The constraints, as in the previous case, produce fails or reduce the sets associated to the values by eliminating the pairs of inconsistent values. These consistence techniques are based on the temporal model defined previously.

2.1 NOTATION, TERMINOLOGY, AND INFORMAL DECRIPTION OF THE CLP(TEMP) LANGUAGE.

In order to define the CLP(Temp) language formally, we should specify each of the four components that characterizes the CLP(X) scheme.

The Σ_{Temp} signature and the INT structure, defined previously, constitute respectively the signature and the structure of the CLP(Temp) instance.

The constraints are represented as atoms as follows.

The unary constraint Ii ∈ Set(Si, Di, Mi) are represented as:

node(Ii, Si', Di', Mi),

where Si' y Di' are the list representations of the sets Si and Di (respectively).

The binary constraint Ii Rel Ij, is represented as:

arc(Ii, Ij, Rel'),

where Rel' is the form of the list corresponding to Rel[4].

For example, if

Rel = {BEF({3,4,5,6,10,11,12}), DUR({2,3,4,20,21,22},{3,4,7,8,9}), AFT({7,8,9}) }

the form of the list corresponding to Rel is

Rel' = [BEF([3,4,5,6,10,11,12]), DUR([2,3,4,20,21,22], [3,4,7,8,9]), AFT([7,8,9])]

The *constraint language* is formed by constraints which have the form:

node(Ii,Si',Di',Mi) and arc(Ii,Ij,Rel'),

besides we include constraints which have the form:

simb_arc(Ii,Ij,Rel'),

where Ii and Ij are identifiers of interval variables, and Rel' corresponds to a general metric-symbolic relation involving symbolic constraints represented as a list.

---

[4]The names *node* and *arc* represent that the unary constraints can be associated to the nodes of a graph, while the binary constraints can be associated to the edges of the graph.

Finally, the *axioms* constituting the theory, are expressed by CH-rules and define the behavior of the CS.
Informally, the CS function consists of eliminating the inconsistent information stored in the constraints which are part of the computational state in the development.

The application of CH-rules transforms an computational state in an equivalent one, by eliminating inconsistencies from the constraints.

The elimination of inconsistencies can occur not only in the unary constraints, but also in the binary constraints. For example, if a computational state includes the constraints:

```
arc(I,J,[BEF([3,4])]),arc(J,K,[BEF([3,4])]),
arc(I,K,[BEF([3,4]),AFT([3,4])])
```

the CH-rules transform the constraint set into:

```
arc(I,J,[BEF([3,4])]),arc(J,K,[BEF([3,4])]),
arc(I,K,[BEF([3,4]))
```

Note that `AFT([3,4])` has been eliminated because is inconsistent respect to the two firsts constraints.
Next we present an example, which have as goal, give an intuitive idea of the behavior that must have the CS of the CLP(Temp) language, and later on we define the CH-rules.
Let's consider the following CLP(Temp) program:

```
p(1).
p(2).
c(A,B,X):-arc(A,B,[BEF([X])]).
```

and the following goal:

```
?- node(I1,[0,1],10,11]),
   node(I2,[10,11],10,21]),
   node(I3,[23,24],10,34]),arc(I1,I2,[BEF([1])]),
   p(X),c(I2,I3,X).
```

We consider the states of the transition system formed just by two components:

- The first one contains the set of subgoals (atoms or constraints) that are not solved yet.
- The second one contains the constraint storage.

Initially, the computational state is:

```
<{node(I1,[0,1],10,11]),node(I2,[10,11],10,21]),
node(I3,[23,24],10,34]),arc(I1,I2,[BEF([1])]),
p(X),c(I2,I3,X)},∅>
```

Since the constraint storage is empty, it is not possible to apply any CH-rule. Next, the subgoal *node(I1,[0,1],10,11)* is chosen, and it is eliminated from the set of subgoals and united to the constraint storage. The next set is:

```
<{node(I2,[10,11],10,21]),node(I3,[23,24],10,34]),
arc(I1,I2,[BEF([1])]),p(X),c(I2,I3,X)},
{node(I1,[0,1],10,11])}>
```

Similarly, the subgoals *node(I3,[23,24],10,34])*, and *arc(I1,I2,[BEF([1])])* are eliminated from the set of subgoals and are included in the constraint storage.
The next state is:

```
<{p(X),c(I2,I3,X)},
{node(I1,[0,1],10,11]),node(I2,[10,11],10,21]),
node(I3,[23,24],10,34]),arc(I1,I2,[BEF([1])])}>
```

The constraint *arc(I1,I2,[BEF([1])])*, included in the constraint storage, allows to apply CH-rules, which must eliminate the inconsistent information and therefore transform the current state in the following state:

```
<{p(X),c(I2,I3,X)},
{node(I1,[0],10,10]),node(I2,[11],10,21]),
node(I3,[23,24],10,34]),arc(I1,I2,[BEF([1])])}>
```

The lists corresponding to the possible starts of I1 and I2 are modified because of the constraint `arc(I1,I2,[BEF([1])])`.

Supposing that more CH-rules can not be applied, the atom p(X) is chosen, and it is searched a clause (fact or rule) in which the head matches p(X). The first clause matching p(X) is p(1). The atom p(X) is eliminated from the current state and the substitution {X/1} holds. The next computational state is the following:

```
<{c(I2,I3,1)},
{node(I1,[0],10,10]),node(I2,[11],10,21]),
node(I3,[23,24],10,34]),arc(I1,I2,[BEF([1])])}>
```

Since the X variable has not been included in any of the constraints, it can not be applied any CH- rule and therefore an atom can be chosen. The only existing atom to choose is c(I2,I3,1). Note that the substitution {X/1} holds. There exists only one clause in which its head matches c(I2,I3,1):

```
c(A,B,X):-arc(A,B,[BEF([X])]).
```

The new computational state is obtained by replacing the body of the clause which unified with that goal by the goal c(I2,I3,1), and it is affected by the substitution that allows the unification:

```
<{arc(I2,I3,[BEF([1])])},
{node(I1,[0],10,10]),node(I2,[11],10,21]),
node(I3,[23,24],10,34]),arc(I1,I2,[BEF([1])])}>
```

The new chosen subgoal is *arc(I2,I3,[BEF([1])])*. It is eliminated from the set of subgoals, and then included in the constraints stored.

```
<∅,{node(I1,[0],10,10]),node(I2,[11],10,21]),
node(I3,[23,24],10,34])
arc(I1,I2,[BEF([1])]),arc(I2,I3,[BEF([1])])}>
```

Starting from this new included constraint, it is possible to apply new CH-rules. If a fail is detected, as occurs because of the inclusion of the last constraint, it backtracks to the following state:

```
<{p(X),c(I2,I3,X)},
{node(I1,[0],10,10]),node(I2,[11],10,21]),
node(I3,[23,24],10,34]),arc(I1,I2,[BEF([1])])}>
```

If other clause matching p(X) is chosen, as p(2), the next computational states are generated:

```
<{c(I2,I3,2)},
{node(I1,[0],10,10]),node(I2,[11],10,21]),
node(I3,[23,24],10,34]),arc(I1,I2,[BEF([1])])}>,

<{arc(I2,I3,[BEF([2])]) ,
{node(I1,[0],10,10]),node(I2,[11],10,21]),
node(I3,[23,24],10,34]),arc(I1,I2,[BEF([1])])}>,
```

```
<∅,{node(I1,[0],10,10]),node(I2,[11],10,21]),
node(I3,[23,24],10,34]),arc(I1,I2,[BEF([1])]),
arc(I2,I3,[BEF([2])])}>
```

Starting from this new included constraint, it is possible to apply new CH-rules. The new computational state obtained by the application of CH-rules is:

```
<{node(I1,[0],10,10]),node(I2,[11],10,21]),
node(I3,[23],10,33]),arc(I1,I2,[BEF([1])]),
arc(I2,I3,[BEF([2])]) >
```

Note that the starts corresponding to the intervals I1, I2 e I3 has been reduced to 10, 11, and 23 respectively. For this case there exists only one solution verifying all of the constraints.

The example has been deliberately simple in order to show the behavior of the CS.

2.2 DESIGN OF CS

The CS acts transforming the constraint set in another one.
The soundness of the CS is proven in [Ibañez,94].
In order to prove the soundness of the CS, it must be proved that both constraint set are logically equivalent wrt the underlying structure. The soundness guarantees that if the constraint set (after applying the transformation) is reduced to the constraint *false*, the initial set of constraint is inconsistent.
If the CS is complete, any inconsistent set of constraints is transformed into the constraint *false*.
If the CS is not complete, the inconsistent set of constraints is transformed into other set of constraints (eventually the same) which is not necessarily the constraint *false*.
A CS complete has the advantage of pruning a priori the derivation tree of the logic program, since every time the CS returns the constraint *false*, it fails, and therefore it avoids to continue trough this branch. However, the cost required by a complete CS can be far greater than the saving avoiding the inconsistent branch (which include a non satisfactory constraint set) of the derivation tree. On the contrary, non complete CS's can be implemented efficiently but they do not allow the same prunes as the case of complete CS do. The greater the completeness degree included in the CS is, the greater the prunes in the tree are.
In this section we define a non complete CS, by taking into account the most important aspects in the design of a CS:

- The cost of the transformation of the constraint set.
- the completeness degree.

The role of the CS consists of transforming the set of accumulated constraints (in the current computational state) into other set of constraint, by eliminating inconsistencies both in the unary constraints and in the binary ones.
The criterion for the design of the CS depends on the election of these transformations. All the CSP techniques (in order to eliminate the inconsistent information associated to the nodes) are possible candidates in the design of these transformations. However, we must be careful in the election of the appropriated mechanism since excess of work in the nodes of the searching tree not necessarily leads to less quantity of comparison.
The technique used to achieve an efficient operational model is based on modifying, not only the information associated to the nodes, but also the information associated to the edges.
The CLP scheme and particularly the CH-rules, provide an appropriated framework in order to perform these tasks. The modification of the information, associated both to the nodes and to the edges, is carried out by the transformations of constraints performed by the CS by the application of CH-rules.

The information associated to the nodes is modified reducing the variables S and D from the constraints: node(I,S,D,Max). The information associated to the edges is modified by reducing the variable R from the constraints art(I,J,R).
The information of the nodes is reduced by applying arc-consistency, when the general metric-symbolic relation contains only one alternative. However in this case, the process is performed by eliminating not only the inconsistent information associated to the nodes, but also the inconsistent information associated to the edges.
The information of the edges is eliminated by applying path-consistency in a symbolic level, that is, not taking into account the characteristic distances.
These processes are called: arc-consistency* and path-consistency* in order to establish a difference respect to the terms used in the literature.

Arc-consistency*.
The following CH-rules eliminate the inconsistent information from both the nodes and the relating edges. These rules are activated only when the general metric-symbolic relation contains only one alternative.

```
1)
node(I,Si,Di,Mi),node(J,Sj,Dj,Mj),arc(I,J,[R]) ⇔
      refine(Si,Di,Sj,Dj,R,Si',Di',Sj',Dj',R',suc)|

 node(I,Si',Di',Mi),node(J,Sj',Dj',Mj),arc(I,J,[R']).
2)
node(I,Si,Di,Mi),node(J,Sj,Dj,Mj),arc(I,J,[R]) ⇔
      refine(Si,Di,Sj,Dj,R,Si',Di',Sj',Dj',R',fail)|
false.
```

The second CH-rule corresponds to the case in which an inconsistency is detected.
The predicate *refine* reduces (in general) the variables Si,Di,Sj,Dj obtaining Si',Di',Sj',Dj' and also reduces the characteristic distance(s) contained in R obtaining R'.
There exists a clause for each basic relation. The constraints for each basic relation are the following:

```
refine(Si,Di,Sj,Dj,BEF(Xbef),Si',Di',Sj',Dj,BEF(Xbef')
,suc ):-
 p([Si,Di,Xbef],[Sj],[Si',Di',Xbef'],[Sj']).

refine(Si,Di,Sj,Dj,BEF(Xbef),Si',Di',Sj',Dj,BEF(Xbef')
,fail).
```

Let L1, ... ,Ln, R1, ... , Rn, L1', ... ,Ln', R1', ... , Rn' be lists of natural number.
The predicate

```
 p([L1,...,Ln],[R1,...   ,   Rn],[L1',...,Ln'],[R1',...,
Rn'])
```

takes the two first arguments as input, and produces the two last arguments as follows:
The list Li' contains the elements li belonging to Li, for which exists elements l1, ... ,li-1, li+1, ... , ln, r1, ... ,rm, belonging, respectively, to the lists
L1, ... ,Li-1, Li+1, ... , Ln, R1, ... ,Rm, such that:

$$\sum_{i=1,n} li = \sum_{i=1,m} ri \qquad (1)$$

The list Ri' contains the elements belonging to Ri', for which exists elements l1, ..., ln, r1, ... ,ri-1, ri+1, ... ,rm, belonging, respectively, to the lists
L1, ..., Ln, R1, ... ,Ri-1, Ri+1, ... ,Rm
verifying the equation (1).
If any of the lists L1', ... ,Ln', R1', ... ,Rm' is empty, the predicate fails.
By the same way the remaining clauses are defined similarly.

Path-consistency*

The path-consistency is performed by CH-rules which represent the composition and the intersection.

<u>Composition</u>

```
    arc(I1,I2,R12),arc(I2,I3,R23) ⇒
         comp(R12,R23,R13')|simb_arc(I,J,R13')
```

<u>Intersection</u>

```
    1)
    arc(I,J,R),simb_arc(I,J,SIMB) ⇔
         inters(R,SIMB,R',suc) | arc(I,J,R').
    2)
    arc(I,J,R),simb_arc(I,J,SIMB) ⇔
         inters(R,SIMB,R',fail)|false.
```

The composition consists of obtaining the symbolic part from the edges (i,k) and (k,j), and it uses a transitivity table for the symbolic relations. This table and that used by Allen are alike although the former uses six basic relations instead of thirteen. The transitivity table is represented by the predicate

`comp_11/3`[5].

For example, the clause corresponding to the relations OVERL, OVERL is:

`comp_11(OVERL,OVERL,[BEF,OVERL]).`

Informally, this clause represents the following formula.

I1 {OVERL} I2, I2 {OVERL} I3 implicates I1 {BEF,OVERL} I3.

Next we present the predicate specifications used in the simplification rules.

<u>Predicate specifications for the composition</u>

comp($\{r_1, ..., r_n\}, \{r_1', ..., r_m'\}, R'$) iif

a) only_simb(r_i, rs_i) ($1 \leq i \leq n$), only_simb(r_j', rs_j') ($1 \leq j \leq m$)

b) $R = \cup R_{ij}$ where comp_11(rs_i, rs_j', R_{ij}) ($1 \leq i \leq n, 1 \leq j \leq m$) (E1)

only_simb(r_i, rs_i) takes r_i (a non disjunctive metric-symbolic relation) as input.

For each pair of non disjunctive symbolic relations rs_1 y rs_2, we include a clause comp_11(rs_1, rs_2, R) where all of its *ground* arguments if the following conditions hold.

a) $\models_{INT}$ (I1 rs_1 I2 $\wedge$ I2 rs_2 I3 implicates I1 R I3

b) There not exists $R' \subset R$ where $\models_{INT}$ I1 rs_1 I2 $\wedge$ I2 rs_2 I3 implies I1 R' I3 (E2)

Since there are 6 different symbolic relations, there is 36 clauses which define the transitivity among the symbolic relations.

<u>Predicate specification for the intersection</u>

Let R and SIMB be *ground* atoms.

a) inters(R,SIMB,R',suc) if $R' = \{r_i \in R : \text{only\_simb}(r_i, rs_i') \wedge rs_i' \in SIMB\} \neq \emptyset$

b) inters(R,SIMB,R',fail) if $R' = \{r_i \in R : \text{only\_simb}(r_i, rs_i') \wedge rs_i' \in SIMB\} = \emptyset$ (E3)

---

[5]The notation *p/n* means that the predicate *p* has arity *n*.

3 FORMAL PROPERTIES OF THE CLP(TEMP) LANGUAGE

The CLP(Temp) language semantic

Let P be a CLP(Temp) program, and G be a goal.

C is a correct answer for the program P and the goal G iif

$$P \models_{INT} (C \rightarrow G)$$

The precise procedural semantic is defined from the construction of the CS by using CH-rules. However, in order to guarantee that the inclusion of the CH-rules does not change the declarative semantic it is necessary to prove that the CH-rules are correct under structure INT.

4 CONCLUSIONS

In this paper we describe a temporal model which allows to unify temporal interval and points, and also symbolic and metric constraints. This formalism uses a unique class of temporal objects (intervals), a unique class of unary constraint affecting an object and a unique class of binary constraint between two objects.

	unific. Between points and int.	CONSTRAINTS									
		UNARY		**BINARY**							
		point	**interv.**	**between points**		**between intervals**			**between points and intervals**		
				symb.	**metr.**	**symb.**	**metr-symb. no disj.**	**metr-symb. disjunct.**	**symb.**	**metr-symb. no disj.**	**metr-symb. disjunct**
Ladkin.	no	no	INDIR.	no	no	DIR.	INDIR	no	no	no	no
Kautz	no	no	INDIR.	no	no	DIR.	INDIR	no	no	no	no
Meiri	no	INDIR.	INDIR.	DIR.	DIR.	DIR.	INDIR.	no	DIR.	INDIR.	no
Tolba	no	no	DIR.	no	no	DIR.	no	no	no	no	no
Proposal	yes	DIR.	DIR.	DIR.	DIR.	DIR.	DIR.	DIR.	DIR.	DIR.	DIR.

Table 3. Expressiveness of the proposal

The table 3 shows the expressive power of proposed model in comparing with the other integration proposals existing in the literature.

The CLP(FD) languages use CSP's consistency techniques.

In order to find solutions in a CSP, two techniques are combined:

- Searching in the tree.
- Elimination or 'filtering' of inconsistent values in each node of the tree.

In the CLP(FD) languages:

- The non determinism, proper of the logic languages, constitutes the searching in the tree.
- The transformation of constraint storage performed by the CS constitutes the 'filtering' of inconsistent values.

The consistency techniques for the CLP(Temp) language are able to eliminate not only the inconsistent information among the temporal objects which constitute the variables of the proposed language, but also the inconsistent information associated to the constraints.

Unlike the other CLP languages, the CLP(Temp) language allows to modify the CS, by adding new CH-rules or modifying the existent ones, therefore the language can be adjusted to particular

characteristics of a determined application. By doing so, efficiency and declarativity may be obtained.
In addition of modifying the CS, which supposes a high degree of understanding by the user, it is possible to extent the language by adding new predicates defined in the CLP(Temp) language, with a high degree of declarativity, and no necessity of knowing deeply all of the capabilities of the CLP(Temp) language.
The CLP scheme has been used in order to define the operational framework, generating the instance CLP(Temp). The CS for this instance has been defined by using CH-rules, which allow to prove soundness formally.

5 REFERENCES

[Allen,83a] J. F. Allen, "Maintaining knowledge about temporal intervals", Commun. ACM 26 (11), 1983.

[Allen,84] J. F. Allen, "Towards a general theory of action and time", A. I. 23(2), 1984.

[Dean,87] T. L. Dean and D. V. McDermott, "Temporal data base management", A. I. 32, pp. 1-55, 1987.

[Decht,91] R. Dechter, Y. Meiri and J. Pearl, "Temporal constraint networks." A. I. 49, pp 61-95, 1991.

[Ibáñez,92a] F. Ibáñez, R. Forradellas, R. Berlanga, F. Barber, G. Martín y F. Toledo, "An efficient method to integrate numeric and symbolical temporal constraints", CHIC ESPRIT PROJECT-5921, Technical Report WP4/T4.1, 1992.

[Ibáñez,92b] F. Ibáñez, R. Forradellas, R. Berlanga, F. Barber, F. Toledo, G. Martín, "Problema de scheduling resuelto mediante un sistema de restricciones temporales", PANEL92, Las Palmas de Gran Canaria, 1992.

[Ibáñez,93] F. Ibáñez, "Integration of Metric and Symbolic Temporal Constraints in CLP", JOINT WORKSHOP ON SPATIAL AND TEMPORAL DATABASES, Coniston, England, 1993.

[Ibáñez,94a] F. Ibáñez, R. Forradellas, R. Berlanga, F. Barber, F. Toledo and G. Martín, "Integrating numerical and symbolical temporal constraints to the CLP scheme", CHIC ESPRIT PROJECT-5921, Technical Report WP4/T4.1, 1994.

[Ibáñez,94b] F. Ibáñez, R. Berlanga, F. Barber and R. Forradellas, "Dos Enfoques de la Programación Lógica con Restricciones", Revista de Informática Teórica y Aplicada, Brasil, (enviado), 4/1994.

[Ibáñez,94c] F. Ibáñez, "CLP(Temp): A formalism to evaluate general temporal constraint", INTERNATIONAL WORKSHOP ON SPATIO-TEMPORAL DATABASES. STDB.94, 1994.

[Kautz,91] H. Kautz and B. Ladkin, "Integrating metric and qualitative temporal reasoning" Proceedings AAAI-91,pp. 241-246, 1991.

[Ladkin,87] P. Ladkin and R. Maddux, "The algebra of convex time intervals", T. R., Kestrel Institute, Palo Alto, 1987.

[Ladkin,89] P. B.Ladkin, "Metric constraint satisfaction with intervals". Technical Report TR-89-038. International Computer Science institute, Berkeley, CA, 1989.

[Ladkin,92] P. B.Ladkin and A. Reinefield, "Effective solution of qualitative interval constraint problems", A.I. 57, 1992.

[MacDe,82] D. V. MacDermott,"A temporal logic for reasoning about processes and plans",. Cogn. Sci. 6, 1982.

[Malik,83] J. Malik and T. O. Binford, "Reasoning in time and space", in: Proceedings IJCAI-83. Karlsruhe,FRG, 1983.

[Meiri,91] I. Meiri, "Combining qualitative and quantitative constraints in temporal reasoning", Proceedings 9th AAAI-91, pp. 260-267, 1991.

[Tolba,93] H. Tolba, "A contribution to the study of temporal reasoning: integrating qualitative-quantitative information and propagation of constraints", IJCAI, 1993.

[Valdés,86] R. E. Valdés-Pérez, "Spatio Temporal reasoning and linear inequalities", A.I. Laboratory , AIM-875, MIT, 1986.

[Vilain,86] M. Vilain and H. Kautz "Constraint propagation algorithms for temporal reasoning", Proceedings of AAAI-86,pp.377-382, 1986.