Distributed (Re)Planning with Preference Information[1]

Pieter Buzing Cees Witteveen

Delft University of Technology,
Faculty EEMCS, P.O. Box 5031, 2600 GA Delft
{p.buzing, c.witteveen}@ewi.tudelft.nl

Abstract

Many planning problems have a distributed nature: different parties have to co-operate in order to make efficient use of shared resources, while not violating any constraints. Additionally, there often are dynamic aspects to be considered. Due to e.g., unexpected changes in the (outside) world or changes of individual goals of the agents each of these actors might come up with a corresponding change of their constraints. As a result, the problem might require some form of replanning of the agent activities such that all constraints can be satisfied.

We discuss a method to deal with changes of temporal constraints in a distributed Simple Temporal Network using preference information. We adapt an existing method to obtain a consistent plan using distributed temporal constraints such that all preferences of all agents above a certain level are satisfied, and changes to the constraints or preferences can be dynamically incorporated, adapting the (inconsistent) plan in such a way that all constraints of some preference level are satisfied.

1 Introduction

Quite a number of real-life planning problems have to do with solving temporal constraints between activities. A well-known approach for solving these temporal problems is to express them as a Constraint Satisfaction Problem (CSP), where variables and constraints between variables are represented in a graph as nodes and edges, respectively. A solution to the CSP is an assignment to the variables such that no constraint relation is violated. One of the simplest temporal problems, the Simple Temporal Planning (STP) Problem (cf. [1]) can be solved in polynomial time using a representation in the form of a labeled directed graph, the so-called *Simple Temporal Network* or STN. In this STN, the nodes are time points and the edges are temporal intervals that restrict the values of the difference between the time points. Recently (cf. [2, 3]), the introduction of preference functions for STNs has been proposed, to indicate in a more expressive way than used in the standard STPs which intervals can be chosen for start/endpoints of activities and how these intervals are preferred by the planner. While in standard STPs it suffices to assign values to variables such that the temporal constraints are satisfied, the

[1]This research is supported by the Technology Foundation STW, applied science division of NWO and the technology programme of the Ministry of Economic Affairs (project DIT5780: Distributed Model Based Diagnosis and Repair).

use of preferences allow us to search for the highest preference level (p-level) at which all constraints can be satisfied.

In this paper we will concentrate on the use of both distributed control for STNs and the use of preference functions and solutions based on p-levels for *replanning* in STNs. The idea is as follows: A solution for an STN can be easily used to derive an *executable plan* for the activities represented. Using preference information, we can search for solutions with a maximal preference value for all the agents. If the constraints in an STN do change, the original solution might become infeasible and a new solution for the changed STN has to be computed, resulting in a new plan. If, however, we use preference information, some changes in the constraints could be accommodated by just letting the actor maintaining the constraint to change his preference function. A new solution then may be found by just letting the agents simply lower their preference level and (re)use an STP-solution from a lower p-level. In this way the consistency of the network (plan) is repaired without too much communication and computation. Moreover, it can be detected in polynomial time whether or not, due to these changes, there still exists an executable plan for the activities represented.

2 STPs and STNs

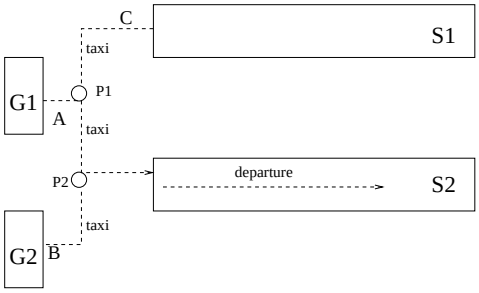
A Temporal Constraint Problem (TCP) consists of a number of time points (corresponding to the start and end points of actions) and a set of binary constraints $a_1 \leq X - Y \leq a_2$ expressing that the difference between the time points X and Y should differ minimally a_1 and maximally a_2 for some $a_1, a_2 \in Z$. These intervals can be used to specify the order in which actions a and a' have to occur, or the minimum amount of delay between a and a'. A Simple Temporal Problem (STP)[1] is a TCP with at most one constraint involving X and Y between a pair (X, Y) of time points. To express that the temporal distance constraint between the beginning and end of an action a is between 10 and 20 time units we denote $X_{a_e} - X_{a_b} \in [10, 20]$.

We will illustrate STPs by using a simple air traffic control (ATC) problem domain to serve as a guiding example. In Figure 1(a) we have an airport with two gates $G1$ and $G2$ and two landing/departure runways $S1$ and $S2$. There are also three aircraft A, B and C: aircraft A is at gate G1, aircraft B at gate $G2$ (they both intend to use runway $S2$ for departure) and C has just landed at landing runway $S1$ and wants to make use of gate $G1$. Obviously, since the airport resources cannot be used by multiple airplanes simultaneously, we need (temporal) constraints on the arrival and departure planning. We assume the aircraft in our example to have (some) autonomy over their planning – effectively being their own planners – and refer to them as agents.

Expressing this example in terms of an STP is not difficult (see Figure 1(b)). Here, the first constraint $X^A_{lG1} - X_0 \in [10, 30]$ specifies that plane A has to leave gate $G1$ between time $t = 10$ and $t = 30$.[2] These constraints are all owned by agent A. Likewise, agent B also has some constraints, as does agent C. The result can be easily represented by a STN (see Figure 2).

A solution to an STP is an assignment of values to each time point X such that all constraints are satisfied. Sometimes such a solution cannot be found and the STP is said to be inconsistent. Checking whether a solution exists can be

[2] X_0 specifies some arbitrary initial point of time



(a) Three aircraft (A, B, and C) at an airport.

$X^A_{lG1} - X_0$	$\in [10, 30]$	$X^B_{lG2} - X_0$	$\in [0, 30]$	$X^C_{lS1} - X_0$	$\in [0, 30]$
$X^A_{aP1} - X^A_{lG1}$	$\in [3, 5]$	$X^B_{aP2} - X^B_{lG2}$	$\in [15, 30]$	$X^C_{aP1} - X^C_{lS1}$	$\in [10, 20]$
$X^A_{lP1} - X^A_{aP1}$	$\in [0, 30]$	$X^B_{lP2} - X^B_{aP2}$	$\in [0, 30]$	$X^C_{lP1} - X^C_{aP1}$	$\in [0, 30]$
$X^A_{aP2} - X^A_{lP1}$	$\in [10, 20]$	$X^B_{aS2} - X^B_{lP2}$	$\in [5, 10]$	$X^C_{aG1} - X^C_{lP1}$	$\in [3, 5]$
$X^A_{lP2} - X^A_{aP2}$	$\in [0, 30]$	$X^B_{aP2} - X^A_{lP2}$	$\in [0, 30]$	$X^C_{aP1} - X^A_{lP1}$	$\in [0, 30]$
$X^A_{aS2} - X^A_{lP2}$	$\in [5, 10]$	$X^B_{bDep} - X^B_{aS2}$	$\in [2, 30]$	$X^B_{eDep} - X^A_{bDep}$	$\in [1, 3]$
$X^A_{bDep} - X^A_{aS2}$	$\in [2, 30]$	$X^A_{eDep} - X^A_{bDep}$	$\in [1, 3]$	$X^B_{aS2} - X^A_{bDep}$	$\in [0, 30]$

(b) The constraints between the actions of A, B, and C.

Figure 1: The constraints for agents A, B and C, where A is at gate G1, B is at gate G2, and C is at runway S1. Here X^A_{lG1}, X^A_{aP1}, X^A_{bDep}, X^A_{eDep} denote "leave gate G1", "arrive at stand P1", "begin departure" and "end departure" respectively. Mind that some constraints involve two agents.

formulated as the well-known *all-pairs shortest path problem*: the STP is consistent if there are no negative cycles (i.e. when constraints imply that the latest possible occurance of an event is *before* the earliest possible occurance) in the all-pairs distance graph, also called the d-graph. Fortunately, this check can be performed in $O(n^3)$ time where n is the number of time points (variables) involved.

3 STPs with Preferences

Sometimes multiple solutions (i.e. instantiations of the variables or time points) to an STP instance may be found while it is not clear which should be regarded as "best". In many practical domains, a solution must not only be feasible, but there is also some kind of *ordering* on the solutions. Khatib ([2]) and Peintner ([3]) present a method that makes use of additional preference information, i.e. an agent that wants to make use of a certain resource not only communicates the time interval(s) that it is aiming for, but also a measure of preference for certain time regions within the interval. Using reasonable preference functions, the constraints with low preference intervals are less tight than the constraints with high preference time regions. This means that starting with a low preference standard is likely to produce a solution (if one exists), after which the preference standard can be increased and a better solution can be sought.

In our example, the constraint $X^A_{lG1} - X_0 \in [10, 30]$ can be extended with a

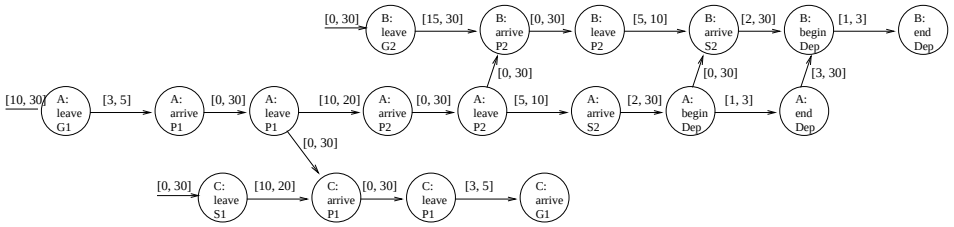


Figure 2: A Simple Temporal Network representing the constraints of Figure 1.

preference function [2] that maps the temporal values in the interval with a preference value. E.g. $X^A_{lG1} - X_0 \in [10, 30]$, $f([10, 15) \cup [21, 30)) = 1$, $f([15, 20)) = 2$, $f([20, 21)) = 3$. This means that agent A has a low preference for the subintervals $[10, 15)$ and $[21, 30)$, a higher preference for the subinterval $[15, 20)$, but the period $[20, 21)$ is considered best. The preference function is defined to be 0 for all values outside of the interval.

The idea now is to use these preferences to derive solutions with maximum minimal preference values. Given a preference range $[0, max]$ such a maximum preference value can be found by applying binary search (cf. [4]) with the temporal constraint intervals $[a, b]$ satisfying $f([a, b]) \geq p$, where p is a preference threshold. The preference function has to be semi-convex to avoid fragmentation of intervals at higher p-levels: This is an STN requirement (cf. [2]). Finding the highest preference value in this way can be done in $O(n^3 \log max)$.

Basically, there are two ways to solve an STP. The first method is to produce the complete d-graph with Floyd-Warshall's all-pairs-shortest-paths algorithm. This process gives a fully connected graph with the shortest paths between all pair of nodes. The more efficient alternative method that we will adopt makes use of triangulation and exploits the fact that not all paths are relevant.

The first step in Xu's $\triangle STP$ algorithm [5] is to produce a triangulated graph of the STN. This produces a graph of triangles G, where the vertices are triples of variables (i.e. $\langle i, j, k\rangle \in G$ iff $(i, j), (i, k), (j, k) \in STN$), and two triangles in G are connected if they share a constraint edge. Consistency checking is done by identifying which triangle(s) an edge (i, j) is in and updating the distance constraints of each of the edges in the triangle. E.g. if a triangle $\langle i, j, k\rangle \in G$ then the interval values are updated following these rules, where $\otimes$ denotes interval composition[3] and $\oplus$ is the intersection of intervals: $I'(i, j) \leftarrow I(i, j) \oplus (I(i, k) \otimes I(k, j))$, $I'(i, k) \leftarrow I(i, k) \oplus (I(i, j) \otimes I(j, k))$, $I'(j, k) \leftarrow I(j, k) \oplus (I(j, i) \otimes I(i, k))$. If one of these interval constraints is assigned a new value, e.g. $I'(i, k) \neq I(i, k)$, then the algorithm performs the same re-calculation on the triangles that contain the edge (i, k). However, if the constraint (e.g. $I(i, k)$) did not change, then this recursive step is superfluous, since the edge (i, j) apparently has no propagation effect on constraints in adjacent triangles. Here lies the pruning power of the $\triangle STP$ algorithm. If during the process of constraint updating an empty interval is produced the STN is inconsistent and the algorithm halts. If no edge labels can be further reduced the STN is minimal and consistent, and an instantiation of the time points can be easily extracted.

[3] $I = S \otimes T$ means that $i \in I$ iff $\exists s \in S, \exists t \in T : i = s + t$. Note that if S and T are both single intervals (as is a typical STP property) then the composition will also be a single interval.

4 Computing Solutions in the Distributed Case

Adapting the $\triangle STP$ algorithm to the distributed case is relatively easy. If one of the nodes in an updated edge (i.e. $I'(i,j) \neq I(i,j)$) belongs to another agent (i.e. $i \notin X^A$ or $j \notin X^A$), then this update has to be propagated to the constraints that are in the other agent's STN. So agent A has to notify the other agent (let's call it agent B) involved of this constraint update: $inform(A, B, i, j, I'(i,j))$. Agent B adds all the triangles that contain edge (i,j) to his Q_T and starts his $\triangle STP$ process. The distributed $\triangle STP$ is shown in Algorithm 1. Let's construct an

Algorithm 1: distributed $\triangle STP$(Agent A, Plan P^A)

1. *consistency* $\leftarrow$ *True*
2. $G \leftarrow$ *Triangulate*(P^A)
3. $Q_T \leftarrow$ *Triangles*(G)
4. **while** $Q_T \wedge$ *consistency* **do**
   - 4.1. $Q_E \leftarrow$ *empty list*
   - 4.2. $\langle i,j,k \rangle \leftarrow$ *First*(Q_T)
   - 4.3. $I'(i,j) \leftarrow I(i,j) \oplus (I(i,k) \otimes I(k,j))$
   - 4.4. **if** $I'(i,j) \neq I(i,j)$ **then**
     - 4.4.1. $I(i,j) \leftarrow I'(i,j)$ *and Enqueue*($(i,j), Q_E$)
     - 4.4.2. **if** $i \in X^{B \neq A}$ *or* $j \notin X^{B \neq A}$) **then** *inform*$(A, B, i, j, I(i,j))$
   - 4.5. $I'(i,k) \leftarrow I(i,k) \oplus (I(i,j) \otimes I(j,k))$
   - 4.6. **if** $I'(i,k) \neq I(i,k)$ **then**
     - 4.6.1. $I(i,k) \leftarrow I'(i,k)$ *and Enqueue*($(i,k), Q_E$)
     - 4.6.2. **if** $i \notin X^{B \neq A}$ *or* $k \notin X^{B \neq A}$) **then** *inform*$(A, B, i, k, I(i,k))$
   - 4.7. $I'(j,k) \leftarrow I(j,k) \oplus (I(j,i) \otimes I(i,k))$
   - 4.8. **if** $I'(j,k) \neq I(j,k)$ **then**
     - 4.8.1. $I(j,k) \leftarrow I'(j,k)$ *and Enqueue*($(j,k), Q_E$)
     - 4.8.2. **if** $j \notin X^{B \neq A}$ *or* $k \notin X^{B \neq A}$) **then** *inform*$(A, B, j, k, I(j,k))$
   - 4.9. **if** $I(i,j)$, $I(j,k)$ *or* $I(i,k)$ *is empty* **then**
     consistency $\leftarrow$ *False*
   - 4.10. **if** *consistency* **then**
     - 4.10.1. **forall** $(m,n) \in Q_E$ **do**
       $T_{m,n} \leftarrow$ *all triangles containing* (m,n)
       forall $\langle r,t,s \rangle \in T_{m,n}$ **do**
       if $\langle r,t,s \rangle \notin Q_T$ **then** *Enqueue*($\langle r,t,s \rangle, Q_T$)
     - 4.10.2. $Q_T \leftarrow$ *Remove*($\langle i,j,k \rangle, Q_T$)
5. *Return consistency*

example with a subnetwork of figure 1(b): $X_1^A = A : leaveP1$, $X_2^A = A : arriveP2$, $X_3^A = A : leaveP2$, $X_1^B = B : leaveG2$, $X_2^B = B : arriveP2$, $X_1^C = C : arriveP1$. See figure 3 for the global STN and the two agent views on this STN, constructed by the agents separately. The graph of agent A is defined by $STN^A = (\{u \in X | (u,v) \in E^A\}, E^A)$, where E^A denotes all the constraints of agent A. Notice that STN^A can contain some "foreign" nodes (i.e. $u \notin X^A$), which are uncontrollable for A, but are involved in his constraint set. Each agent starts with interval values $[-\infty, \infty]$ for edges that he has no information about, e.g. $I(0, B2)$ in agent A's network. During the construction of the minimal STP^A agent A calculates

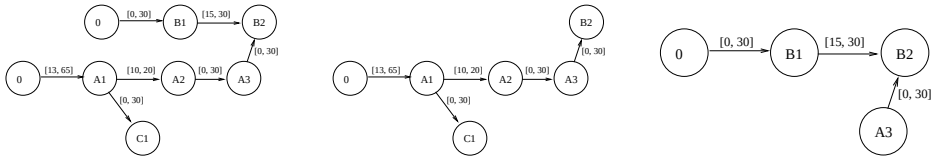


Figure 3: A part of the STN (a), separated in agent A's constraint network (b) and agent B's constraint network (c). Mind that they have a common constraint ($I(A3, B2)$) and some common nodes: X_0, X_3^A, and X_2^B.

$I'(0, B2) = [23, 145]$[4] and as both $I'(i, j) \neq I(i, j)$ and $B2 \notin X^A$ this new value $I'(0, B2)$ is sent to agent B.

Agent B had already calculated $I(0, B2) = [15, 60]$, but this edge is assigned the new constraint $[23, 145] \oplus [15, 60] = [23, 60]$. Following the steps in the $\triangle STP$ algorithm, agent B now considers all triangles effected by the $I(0, B2)$ update. The edge $(0, A3)$ will be updated to a new constraint interval $[-15, 60]$ and because $A3 \notin X^B$ an $inform(B, A, 0, A3, [-15, 60])$ is sent out.

Agent A reconsiders the edge $(0, A3)$ and assigns it the constraint value $[-15, 60] \oplus [23, 115] = [23, 60]$. Through this process all constraints are propagated and if no agent finds an inconsistency (i.e. an empty interval) the global STN is consistent.

4.1 Incorporating Preference Levels

Until now we have only discussed constraints as 1-dimensional time intervals, without any further preference information. It is clear however, that there is some slack in the STN, so improvements might be possible. Recall that we define the preference function $f([a, b]) \in [0, max]$ for each constraint. In our example these constraint preferences might be these:

$$X_1^A - X_0 \in [10, 110], f([10, 13) \cup [65, 110]) = 1, f([13, 15)) = 3, f([15, 65)) = 2$$
$$X_2^A - X_1^A \in [5, 30], f([5, 10) \cup [20, 30)) = 1, f([10, 15) \cup [16, 20)) = 2, f([15, 16)) = 3$$

We now have different sets of constraints, one for each preference level. The question now is: what is the highest minimal preference level that still produces a consistent STN? An efficient method to reach an STN with the highest preference of all agents is through binary search. In our example we would initiate the search process at preference level $p = 2$, so we use the distributed $\triangle STP$ algorithm discussed in the previous subsection to determine whether all constraints (a, b) with $\forall t \in I(a, b) : f(t) \geq p$ produce a consistent STN. Since these level-2 intervals are the same as the ones used earlier in this section to produce a consistent STN we can conclude that a level-2 STN is solvable.

The next step is to increase the preference level to $p = 3$ and see if an even better network can be found. These constraints are tighter than before: $X_1^A -$

[4]Essentially $I'(0, B2) \leftarrow I(0, B2) \oplus (I(0, A1) \otimes I(A1, A2) \otimes I(A2, A3) \otimes I(A3, B2))$.

$X_0 \in [13, 15]$, $X_2^A - X_1^A \in [15, 16]$, $X_3^A - X_2^A \in [6, 10]$, $X_1^B - X_0 \in [15, 18]$, $X_2^B - X_1^B \in [20, 24]$, and $X_2^B - X_3^A \in [10, 20]$

The distributed $\triangle STP$ algorithm will notice an inconsistency when agent A calculates $I'(0, B2) = I(0, B2) \oplus (I(0, A3) \otimes I(A3, B2)) = [23, 60] \oplus ([34, 41] \otimes [10, 20]) = [44, 61]$ and forwards this to agent B, who cannot fulfill this constraint since he calculated $I'(0, B2) = I(0, B2) \oplus (I(0, B1) \otimes I(B1, B2)) = [23, 60] \oplus ([15, 18] \otimes [20, 24]) = [35, 42]$. Both agents will find an empty interval, indicating an inconsistency.

The preference level has to be adjusted now to a lower level. In our example, we return to $p = 2$, for which all agents already found a feasible solution. In general, the distributed $\triangle STP$ algorithm has to be performed at most $\log max$ times if the preference function gives values from the range $[0, max]$.

5 Changing Constraints and Repairing Solutions

If one or more agents change their temporal constraints, they might be forced to reconsider their current plan based on the solution of the common distributed STP. Most common changes in temporal constraints concern the *tightening* or the *relaxation* of constraints. For temporal constraints specifying intervals, such operations come down to *contraction* or *dilatation* of an interval $[a, b]$ to an interval $[a', b']$ where either $a' < a$ and $b' > b$ or $a' > a$ and $b' < b$ holds. Together with this interval transformation, we have to consider the transformation of the associated preference function f. Any transformation of f to f' that satisfies the semi-convexity condition for f' can be handled by the methods described below. We will now only discuss the consequences for replanning of interval contraction, since dilatation generally relaxes the constraints, possibly leading to new (and better) solutions, but not posing a threat to the plan execution.

A contraction of the preference function between time points X and Y transforms the interval $I(X, Y)$ to $I'(X, Y)$ and in particular it changes $I_p(X, Y) = [x \in I(X, Y) | f(x) \geq p]$ to $I'_p(X, Y)$ for preference level p. Let $d_p(X, Y)$ denote the (minimal) constraint between X and Y at preference level p in the network, and if the network is consistent for preference level p then of course $d_p(X, Y) \subseteq I_p(X, Y)$. Three cases need to be distinguished now. (Notice that in all scenarios the agents don't have to start the planning process from scratch, but can make use of the current STN.)

Firstly, if $d_p(X, Y) \subseteq I'_p(X, Y)$ then the constraint clearly is still satisfied and the network is still consistent. Nothing needs to be changed or communicated.

Secondly, if the new preference interval only partially overlaps with the global constraint network can be re-established by two methods: a) by restricting the constraint to $d'_p(X, Y) = d_p(X, Y) \oplus I'_p(X, Y)$ and restarting the distributed $\triangle STP$ algorithm with $Q_T \leftarrow$ all triangles containing (X, Y); b) by searching for the lowest p' value that can fulfill $d_p(X, Y) \subseteq I'_{p'}(X, Y)$. The first option maintains the optimal preference level at the cost of some communication and further restriction of other agent's constraints. The alternative is to accept the preference loss: the agent might not achieve the highest preference value, but saves on computation and communication effort. If such a value p' can not be found then only option a) is feasible.

Thirdly, if a contraction of the preference function results in two disjoint sets ($d_p(X, Y) \cap I'_p(X, Y) = \emptyset$) this implies that at the current p-level no solution can be found: i.e. not all agents can be assigned time values of preference p. The

agent again has basically two options: a) he checks his history for a preference level $p' < p$ that satisfies $d_p(X,Y) \cap I'_{p'}(X,Y) \neq \emptyset$, i.e., is there a preference level p' for which the agent knows that the global constraints can be met by other agents with minimal preference still p; b) he can use his history to find a p' that makes a solution possible at the global preference level p': $d_{p'}(X,Y) \cap I'_{p'}(X,Y) \neq \emptyset$. The first option does not require a global preference fall-back, so only the new constraint between X and Y needs to be propagated. The second strategy does call for a global preference return and possibly a number of iterative $\triangle STP$ loops to find the optimal preference level. In both cases the agent communicates to others the updated $d_{p'}(X,Y) = d_{p'}(X,Y) \oplus I'_p(X,Y)$ and will initiate the distributed $\triangle STP$ algorithm with $Q_T \leftarrow$ all triangles containing (X,Y).

Let us illustrate a preference constraint contraction of the second type with an example. Agent A changes his preference function for the interval between X_1^A and X_2^A to the following: $f([8,12) \cup [19,22)) = 1, f([12,15) \cup [16,19)) = 2, f([15,16)) = 3$. This poses a threat for the plan quality, since the constraint that assures a $p = 2$ solution is $X_2^A - X_1^A \in [10,17]$ –this can be deduced via triangulation– and agent A's new preference interval can no longer contain (and satisfy) it: $[10,17] \not\subseteq [12,19]$. The agent will update the edge (X_1^A, X_2^A) to the value $[10,17] \oplus [12,19] = [12,17]$ and restart the $\triangle STP$ algorithm with $Q_T \leftarrow$ all triangles containing (X_1^A, X_2^A).

6 Conclusion

We have discussed the adaptation of an efficient algorithm for computing solutions for an STN to distributed cases and the use of preference functions. We have shown that the use of preference information can be easily used to accommodate for small changes in the network, e.g., if some agents change their constraints. In such cases, often it suffices to re-use previous computations for obtaining preference levels for solutions of the original network.

References

[1] R. Dechter, I. Meiri, and J. Pearl. Temporal constraint networks. *Artificial Intelligence*, 49:61–95, 1991.

[2] L. Khatib, P. Morris, R. Morris, and F. Rossi. Temporal constraint reasoning with preferences. In *17th International Joint Conference on AI*, pages 322–327, 2001.

[3] B. Peintner and M. E. Pollack. Low-cost addition of preferences to dtps and tcsps. In *the 19th National Conference on Artificial Intelligence*, July 2004.

[4] F. Rossi, K. Venable, L. Khatib, P. Morris, and R. Morris. Two solvers for tractable temporal constraints with preferences. In *Proc. AAAI 2002 workshop on preference in AI and CP*, Edmonton, Canada, July 2002.

[5] L. Xu and B. Y. Choueiry. A new efficient algorithm for solving the simple temporal problem. In *TIME-ICTL*, pages 212–222, Cairns, Queensland, Australia, 2003. IEEE Computer Society Press.