Ann Oper Res (2007) 155: 391–415
DOI 10.1007/s10479-007-0207-z

Handling fuzzy temporal constraints in a planning environment

Marc de la Asunción · Luis Castillo · Juan Fernández-Olivares · Oscar García-Pérez · Antonio González · Francisco Palao

Published online: 13 July 2007

Abstract An interleaved integration of the planning and scheduling process is presented with the idea of including soft temporal constraints in a partial order planner that is being used as the core module of an intelligent decision support system for the design forest fire fighting plans. These soft temporal constraints have been defined through fuzzy sets. This representation allows us a flexible representation and handling of temporal information. The scheduler model consists of a fuzzy temporal constraints network whose main goal is the consistency checking of the network associated to each partial order plan. Moreover, we present a model of estimating this consistency, and show the monitoring and rescheduling capabilities of the system. The resulting approach is able to tackle problems with ill defined knowledge, to obtain plans that are approximately consistent and to adapt the execution of plans to unexpected delays.

1 Introduction

Artificial Intelligence Planning techniques have shown very useful in the resolution of different problems related to logistics and workflow domains, just to cite a few (Biundo et al. 2003). However, many real world problems require the inclusion of an implicit representation and handling of time for the plans to be correctly executed so that some scheduling techniques must be imported into a planning framework. But on the other hand, many realistic scheduling problems still require the inclusion of some planning techniques to work properly (Bartõk and Mecl 2003). The work presented in this paper is based on one of these interdisciplinary domains devoted to mobile workforce management in crisis situations: the SIADEX project (de la Asunción et al. 2003, 2005). lthough the techniques being developed in this project are devoted to the assisted design of forest fire fighting plans, they are

This work has been partially supported under the project MCyT TIC2002-04146-C05-2 and the contract NET033957 with the Andalusian Regional Government.

M. de la Asunción · L. Castillo (✉) · J. Fernández-Olivares · O. García-Pérez · A. González · F. Palao
Dpto. Ciencias de la Computación e Inteligencia Artificial, ETSI Informática, Universidad de Granada, 18071 Granada, Spain
e-mail: L.Castillo@decsai.ugr.es

general enough so as to be of application to other crisis management domains either in military frameworks (operations planning (Wilkins and Desimone 1994), air campaign design (Myers 1999) or noncombatant evacuation operations (Munoz-Avila et al. 1999)) or in civil frameworks (oil spills (Bienkowski 1995), floods (Biundo and Schattenberg 2001) or forest fires (Avesani et al. 2000; Cohen et al. 1989)). In these real domains one must consider many different techniques but this paper focuses on two of them that are particularly relevant.

On the one hand, there is a need to integrate planning and scheduling techniques so that intelligent decision support systems are able not only to reason about action and changes, but also about time and resources.

On the other hand, in order to solve this kind of problems it is necessary to consider a flexible representation and handling of time. Temporal planners (Bacchus and Ady 2001; Do and Kambhampati 2001; Haslum and Geffner 2001; Smith and Weld 1999) use a rather rigid notion of time in the sense that time and durations are assigned and compared taking into account only strict equality. Hence these temporal planners may only be applied to problems where temporal knowledge is sharply defined and resulting temporal plans have to be executed exactly in the time line defined by the planner since, otherwise, the plan will fail. Let us suppose that in a forest fire episode, two fire fighting brigades are expected to arrive to the waiting area exactly at 10:00 am, but they arrive at 10:05 am and 10:10 am respectively. Is the plan still valid?

In many real applications the temporal bounds of activities, like deadlines or durations, are implicitly considered as flexible bounds at some extent, that is, they are not a rigid matter because of incomplete or vague knowledge or preferences, unpredictable behaviors or execution errors. Therefore this paper presents MACHINE^{TF}, an extension of a previous work of the authors (Castillo et al. 2001), based both on the use of an interleaved integration of the planning and scheduling processes and on the representation and handling of soft temporal constraints. For example, when one makes a plan in order to transport by truck a fire fighting brigade to a forest fire, one doesn't know the exact time needed to activate the members of the brigade, nor the exact time that the drive takes to complete the route by truck, nor the exact time of arrival at the waiting area. Instead, all these temporal constraints are roughly defined, but these plans can easily be designed by an expert. So this paper explains how to represent these soft temporal constraints and how to include successfully soft constraint handling procedures into a planning framework.

Soft constraints have been defined by means of fuzzy sets (Dubois et al. 2003), a very expressive formalism to represent imprecision and preference of the user when he/she uses soft temporal constraints. Hence, the basic representation of the temporal knowledge will be a Fuzzy Temporal Constraint Network (FTCN) (Vila and Godo 1994b; Marín et al. 1997). This also allows the planner to build plans based on vague temporal constraints of the form: "the action of driving the truck will take about 100 time units" or "action a must be delayed more or less between 20 and 30 time units after action b". This type of "soft constraints" provide several advantages:

- It is possible to model domains in which temporal knowledge is vaguely defined and to obtain temporal plans of practical use for these domains.
- It becomes a very interesting approach in time critical problems, like crisis management, space missions or complex scheduling problems, where execution times are very tight and perhaps a solution that completely meets the temporal constraints is not possible and, instead, a relaxation of the constraints is feasible and an approximate solution could be found.

- A temporal plan built on top of a FTCN is not a single solution, but a class of possible solutions, i.e., a set of solutions for which the set of actions is always the same, but their temporal interleaving might slightly change from each other. In fact, slightly different temporal schedules of the same set of actions might be obtained by a solution extraction procedure as explained in the paper. This flexible representation of plans would allow to adapt to possible delays during the execution of the plan without the need to re-plan a new sequence: let us suppose that a sequence of actions has been designed and that it has already started its execution, if an unexpected delay occurs, it is still possible to obtain a new time line for the remaining of the plan just by changing to an alternative schedule able to adapt to the existing delay.

The basic process of MACHINE^{TF} is a partial order model in which at every step the plan being designed by the planner is passed to the scheduler module in order to determine its consistency. An inconsistent plan is immediately detected, even prior to complete all its actions, and rejected. In the case of consistent plans, since the consistency in a FTCN is a degree matter the degree of temporal consistency is used by the heuristic of the partial order planner in order to search plans appropriate to the preferences of the user. The problem of an efficient estimation of the consistency is addressed through the calculus of optimistic and pessimistic bounds of the consistency.

The paper is organized as follows. It does not discuss the domain of crisis management (forest fires) in detail, instead, it is focused on the techniques underlying the approach to integrate planning and scheduling and to represent and handle soft temporal constraints. For a more general description of SIADEX, see de la Asunción et al. (2003, 2005). After giving a description of the integration of planning and scheduling we have used in Sect. 2, we first describe in Sect. 3 how the fuzzy sets are used as a model of knowledge representation, and then turn in Sect. 4 to the description of the scheduler process based on the use of FTCN and the problem of estimation of the consistency for these networks is addressed. Section 5 shows the monitoring and rescheduling capacities of MACHINE^{TF} and an example of the process. Section 6 presents a monitoring example and Sect. 7 concludes.

2 Integration of planning and scheduling

The main goal of this work is the representation and handling of soft temporal constraints in a partial order planner. Moreover, we are interested in the application of the model to real problems, like the assistance to experts in the task of designing forest fire extinction plans described above in the context of the SIADEX project. Therefore, a flexible enough handling of time in the complete process is needed.

Thus, in order to successfully deal with temporal constraints, the planning algorithm should be extended to cope with some temporal reasoning capabilities that traditionally belong to scheduling systems like constraint posting and propagation, consistency checking or solution extraction. In the literature, there are mainly two families of architectures for the integration of planning and scheduling. On the one hand (Fig. 1) a planner and a scheduler execute one after the other in a waterfall model. This model is easier to implement but has many drawbacks mainly related to backtracking points between both isolated systems.

Instead, we have followed an interleaved integration (Fig. 2) where part of the decisions of the scheduler have been introduced intimately into the planning engine.

The main idea of this integration model is to allow the collaboration between the planner and the scheduler in order to search the best plan for the problem. After every step of the planning engine the current plan under construction is sent to the scheduler. It checks the

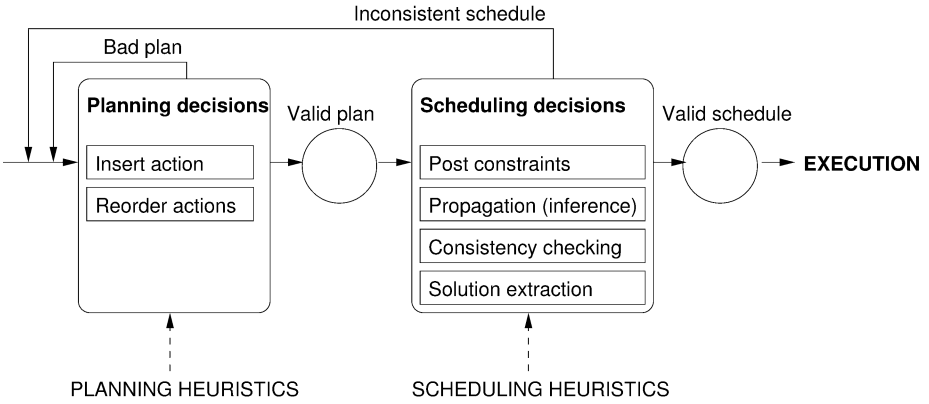


Fig. 1 A batch model of planning and scheduling

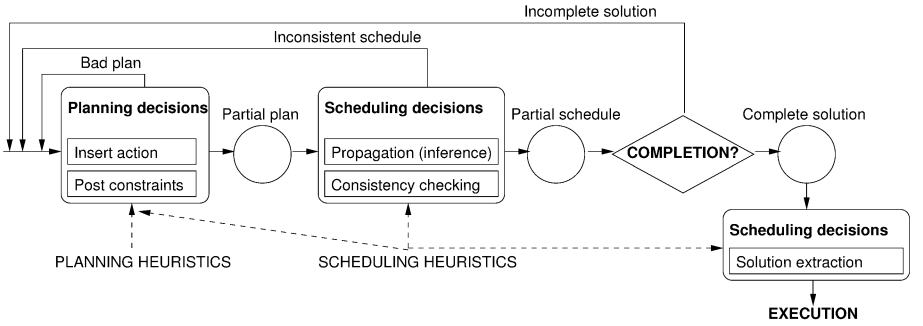


Fig. 2 An integrated model of planning and scheduling

temporal consistency of the constraints associated to this current plan, and this information is used by the planner to reject the plan, in the case of inconsistency, or to follow with the refinement of this plan, in the case of a consistent plan. Therefore, the integration between planner and scheduler overcomes the backtracking problems of the former approach so that, for example, an inconsistency of the constraints makes the planner to backtrack immediately.

Another important point is the inclusion in the partial order planner of temporal information in the way of soft temporal constraints. Some examples of these constraints are the statement of a makespan, the duration of actions, the temporal distance between pair of actions or deadlines goals.

In the next section we propose an extension of the knowledge representation of our planner MACHINE (Castillo et al. 2001), a partial order causal link based planner (Weld 1994), able to deal with soft temporal constraints.

3 Extending the knowledge representation

The starting point to be addressed when extending a planning model to account for new knowledge is its model of actions, that is, the way the planning framework represent changes in the world. Other issues to be defined are the definition of problems and the definition of

plans. In this case, we have extended the basic model of action of our planner MACHINE (Castillo et al. 2001) (basically the same than PDDL 2.1 (Long and Fox 2003b)), to allow the representation of fuzzy temporal constraints.

3.1 Representing and handling fuzzy temporal constraints

Planning for real problems is nothing without a valid execution of the plan, and this often implies the need to deal with temporal constraints if one wants to obtain a realistic solution. There are several examples in the literature that have defined approaches to deal with these constraints (Do and Kambhampati 2001; Haslum and Geffner 2001; Smith and Weld 1999), however, in many real problems, some (or many) of these constraints are not rigidly defined. This lack of rigidity appears in several different ways.

Let us consider that we have designed a plan to carry a fire fighting brigade from one location to another including transportation and loading and unloading their tools. Let us suppose that the maximum duration for the unloading operation is 60 time units and the whole makespan is 240 time units maximum. A rigid interpretation of temporal constraints would imply a chain of execution failures if the unloading of tools and material takes 61 time units and the makespan finally grows up to 243 time units, and also, in the case of more complex plans, it will surely raise important questions about the causal correctness of the remaining plan. In real life, and depending on acceptance criteria, this relative delay could be acceptable if nothing else can be done and the goal is finally achieved. Perhaps an interval based representation could be appropriate and we could accept that the duration of the unloading could be represented like $[60 - \delta_1, 60 + \delta_1]$ and the makespan like $[0, 240 + \delta_2]$ where δ_1, δ_2 depend on acceptance criteria. This interval based representation has been used in the literature as a valid means to deal with temporal imprecision or temporal preference in planning systems (Laborie and Ghallab 1995; Muscettola 1994), however it presents some drawbacks that need to be clarified. Let us suppose in the previous example that $\delta_1 = 2$ and $\delta_2 = 4$. This would make the execution of the previous plan acceptable, but it would also allow the planner to accept a longer plan with a makespan of say 244.

However, this does not reflect exactly the desires of the user when he relaxed the restrictions. A relaxation of the temporal constraints is not only a widening of the bounds of the constraint, but also a distribution of his preference. In the example, he accepts a makespan between 240 and 244 but, and this is the most important, not all of them are equally preferred, the planner should give priority to plans whose makespan is closer or under 240 time units. At this point it is clear that an interval based representation is not able to represent this information and, therefore, it does not seem expressive enough to represent appropriately the semantics of a relaxation of the constraints in a planning and scheduling framework.

In order to solve this representation problem, we propose the use of fuzzy intervals to model the concept of soft temporal constraints in a planning and scheduling framework as a more expressive formalism able to represent the preferences of the user when he relaxes a temporal constraint, so that the initial constraint represents what the user *desires*, with the highest priority, and the relaxed constraint represents what the user *would admit* in the case that his desires cannot be satisfied and giving priority to the values closer to the initial satisfiability. A fuzzy interval, like that shown in Fig. 3a, is able to represent this information by means of its membership function μ, i.e., the degree of satisfaction of a temporal constraint over the time line (Dubois et al. 1993). Initial satisfiability is represented as the set of

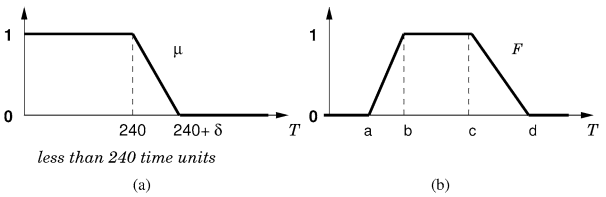


Fig. 3 Two fuzzy temporal intervals

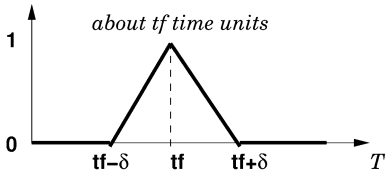


Fig. 4 A fuzzy temporal constraints representing "about t_f time units"

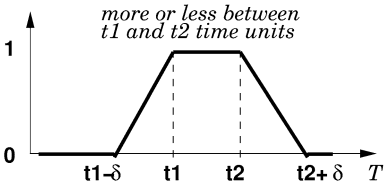


Fig. 5 A fuzzy temporal constraints representing " more or less between t_1 and t_2 time units"

values t such that $\mu(t) = 1$, that is, the strict desires of the user. The soft constraint is represented as the set of values t such that $0 < \mu(t) < 1$, i.e., those values that are admissible and that are ordered in preference to the initial satisfiability, and finally, those values t such that $\mu(t) = 0$ are not admissible at all, that is, the user cannot admit these values in any case. Therefore, the preferences of the user regarding a relaxed and flexible temporal constraint C are perfectly represented by the following membership function

$$\forall t \in T, \quad \mu_C(t) = \alpha,$$

where $\alpha \in [0, 1]$ represents the degree of user satisfaction of the constraint C. This model of soft temporal constraints implies a certain degree of controllability of the duration of actions, since during their execution, the executive (human or computer) may be required to shorten or enlarge its duration to better fit the overall temporal constraints. However, the model is also subject to the occurrence of exogenous events that might modify the duration of an action. Exogenous events are not controllable, that is, they fall outside of the scope of the executive, but the approach described in this paper is able to monitor these exogenous changes and to suggest the executive to modify the duration of subsequent actions in order to accommodate to possible delays while maintaining consistency.

This idea of constraint relaxation is well known in many constraints satisfaction problems approaches (Dechter 2003) and it is the main motivation of this paper for dealing with fuzzy sets. In particular, the remaining of this paper assumes the use of convex trapezoidal fuzzy subsets like the fuzzy subset F shown in Fig. 3b that will be noted as tuple with their respective points $F = (a, b, c, d)$. Obviously, this fuzzy notation may also be used to represent crisp subsets, i.e. subsets with no relaxation at all, for example, the tuple (b, b, c, c) represents the classic strict interval $[b, c]$.

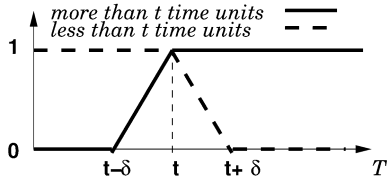


Fig. 6 Two fuzzy temporal constraints representing "more than t time units" or "less that t time units"

In particular, the different types of fuzzy temporal constraints that will be used are shown in Figs. 4, 5 and 6, where in all the cases the value δ encodes the admissibility limits for each constraint, and it may also be different for each side of the fuzzy set.

3.2 Actions, preconditions and effects

An action a in MACHINE^{TF} is represented taking into account the following issues.

- It is represented by means of two time points, namely $start(a)$ and $end(a)$.
- An action has effects, that is, changes in its environment and every effect takes some time to be achieved. This time may not be precisely known, so it is represented as the fuzzy temporal constraint "about t_f time units" (Fig. 4) or "more or less between t_1 and t_2 time units" (Fig. 5) that must be between $start(a)$ and $end(a)$. This knowledge represents a flexible constraint where the initial constraint has been relaxed through a parameter δ. Obviously, if the delay is strictly bounded and the user doesn't permit the relaxation then $\delta = 0$.
- An action has preconditions, that is, conditions that have to be made true by the effect of another action prior to its execution. Since effects take some time to be achieved this is one of the main source of (soft) temporal constraint posting between actions. Let us consider that the condition f of action a is satisfied by the effect f of action b that takes "about t_f time units" to be achieved. Then there must be a (soft) temporal constraint that enforces $start(a)$ to be "more than t_f time units" after $start(b)$ (Fig. 6).
- Actions have a duration that may be either unlimited (although it is calculated during the planning process) or bounded, that is, they may have a maximum duration ($maxbound$) allowed specified in terms of "less than $maxbound$ time units" (Fig. 6). All the actions have an minimum duration, that is, the duration required to obtain all its effects.

Then, the domain of a planning problem is the set of actions available to solve that problem.

Example 1 This example, inspired in zeno problems of the international planning competition (Penberthy and Weld 1994; Long and Fox 2003a), presents a transport problem in a crisis situation. The problem, depicted in Fig. 7 consists in carrying two fire fighting brigades from their respective bases (i.e., city-a and city-b) to the crisis scenario (city-c). In order to do that, a helicopter able to transport one brigade must be used. This helicopter has its base at city-a and must pick up brigade-1 and brigade-2 and carry them to city-c. The helicopter may fly at two different speeds: a slow speed (named fly) that takes "more or less" 10000 time units to travel from city-a to city-c and a fast speed (named zoom) that takes "more or less" 7000 time units. In addition to this, brigade-1 must reach city-c before 7500 time units from the begining of the plan (fuzzy temporal reference (0, 0, 7500, 7600)), but brigade-2 does not have any deadline.

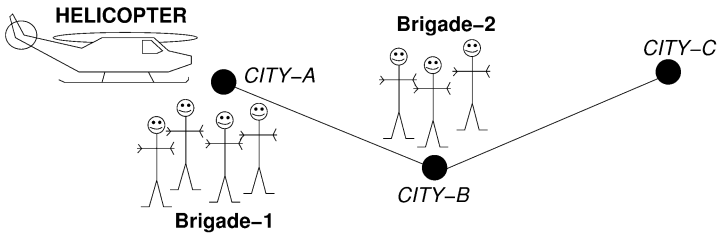
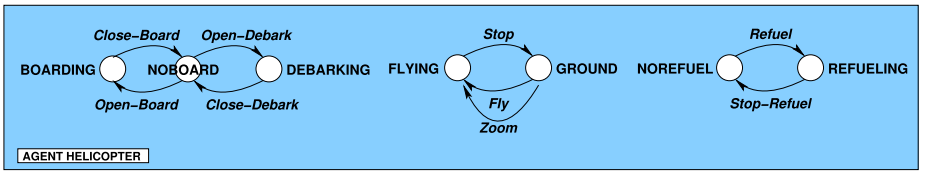


Fig. 7 A transport domain inspired in the domain Zeno




```
(:agent helicopter1
   (:states (ground flying norefuel refueling
             noboard boarding debarking))
   (:action fly
    :parameters (?city1 ?city2)
    :max-duration (6900 7000 7000 7100)
    :pre-condition
       (and (helicopter-at helicopter1 ?city1)
            (refueled helicopter1)
            (state helicopter1 ground))
    :sim-condition
       nil
    :effects
       (and ((not (helicopter-at helicopter1 ?city1)) (0 0 0 0))
            ((not (state helicopter1 ground)) (0 0 0 0))
            ((not (refueled helicopter1)) (6900 7000 7000 7100))
            ((helicopter-at helicopter1 ?city2)(6900 7000 7000 7100))
            ((state helicopter1 flying) (0 0 0 0))))
   (:action refuel
    :parameters nil
    :max-duration (75 100 100 225)
    :pre-condition
       (state helicopter1 norefueling)
    :sim-condition
       (state helicopter1 ground)
    :effects
       (and ((not (state helicopter1 norefuel)) (0 0 0 0))
            ((refueled helicopter1) (75 100 200 225))
            ((state helicopter1 refueling) (0 0 0 0))))
```

Fig. 8 The structure and part of the domain of the travel domain of Example 1. All the available actions for the helicopter are shown. Note that there are two modes of flight depending on their speed: a fast flight (zoom), that takes shorter, and a slow flight (fly), that takes longer

In this case, the domain representing all possible actions is represented in Fig. 8. The operation of the helicopter is represented as a finite state automaton at the top of the figure, showing all possible states for the helicopter. The figure also shows the representation of actions fly and refuel with their respective preconditions and effects.

As can be seen, the helicopter must fly at its maximum speed to meet the given constraints to reach city-c since flying at the minimum speed would be too slow to meet the constraint imposed on the transportation of brigade-1. This is depicted in Fig. 9 where the initial part

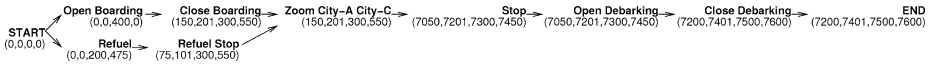


Fig. 9 A soft temporal plan for part of problem of Example 1 under the deadline constraint (0, 0, 7500, 7600) for brigade-1 to be at city-c. Every action is labeled with its respective execution time as a fuzzy temporal reference

of the plan devoted to carry brigade-1 is shown. There may be seen that the refuelling action may be executed in parallel with the boarding of the brigade and its equipment and also that the plan finally ends a bit earlier than required (7200, 7401, 7500, 7600), that is, the earliest ending time is 7401 time units although, in the case of an unexpected delay, it could be delayed up to 7500 time units with the highest possibility degree.

3.3 Problems

Problems are stated like a conjunction of subproblems (literals) that must be solved, so the planner must design a sequence of actions from the domain that achieve these subproblems. In the domain of Example 1 a possible goal would be `(at brigade-1 city-c)` to find a plan for a trip from `city-a` to `city-c`. However, a problem may also include soft temporal constraints.

- Deadlines. Some of the literals have to be achieved at a given time. Deadlines are expressed as a soft constraint in any of the forms shown in Figs. 4, 5 or 6. For example the deadline goal

  `(at brigade-1 city-c) AT (11pm-δ,11pm,11pm,11pm+δ)`

  might be used to require `Brigade-1` to arrive in `city-c` "more or less at 11 pm".
- Makespans. The total length of the plan may also be restricted by means of a soft temporal constraint like any of the ones shown in Figs. 4, 5 or 6. An example could be to find a plan with "approximately less than or equal to 7500 time units".

3.4 Plans

Plans are a partially ordered sequence of actions and in MACHINE^{TF} they are deployed over a Fuzzy Simple Temporal Constraint Network.

Definition 1 A FTCN $\mathcal{N} = \langle X, C \rangle$ is composed of a set of variables $\mathcal{X} = \{X_0, X_1, \ldots, X_{n+1}\}$ and a set of fuzzy binary temporal constraints defined between them $\mathcal{C} = \{C_{ij} | 0 \leq i, j \leq n+1\}$ (Marín et al. 1997).

Every variable X_i is a crisp variable whose domain is the real time scale T. Variables X_0 and X_{n+1} are two dummy variables used to represent the beginning and the end of the network. Every fuzzy binary constraint C_{ij} restricts the possible relative values of X_i and X_j, i.e., $X_j - X_i \leq C_{ij}$. Every constraint C_{ij} is defined as any of the types shown in Figs. 4, 6 or 5 is represented by a possibility distribution π_{ij} over the continuous time scale T. In the framework of MACHINE^{TF}, a FTCN is used to represent a fuzzy temporal plan where every variable represents the execution time of one of the actions of the plan and every fuzzy binary temporal constraint is posted during the resolution process every time that an action solves a subgoal or when actions are reordered to avoid interferences between them, as will be explained later. Figure 9 shows a soft temporal plan where the annotations under each action a represent the soft constraint defined for $start(a)$, i.e., $C_{start(a),0}$ in the FTCN.

4 The scheduler

The scheduler module we used in Fig. 2 takes as input a partial plan. This partial plan is the current plan under construction by the planner, that can be even an plan with some actions missing, and the main idea is to check the temporal consistency of this plan. That is, the main goal of the scheduler is to study if there are solutions to the constraints set associated with the partial plan. Therefore, in a first step the scheduler generates the FTCN of the partial plan, analyzes and propagate all the constraints, and finally it gives as output information about the consistency of such network.

4.1 Fuzzy temporal constraint network concepts

In this process it is very important to take into account the fuzzy representation of the temporal information we have used. Thus, we can see how in a soft temporal plan, like the one shown in Fig. 9, we don't represent a unique solution but a set of solutions such that all of them have the same actions, but they might be scheduled for execution in different orders, all of them consistent with the FTCN of the temporal plan. This is very important since during the design of the plan, MACHINE^{TF} does not commit to assign a precise execution time for every action like most temporal planners (Do and Kambhampati 2001; Haslum and Geffner 2001; Smith and Weld 1999). This excess of commitment would make the planner to increase the number of backtracks. Instead, MACHINE^{TF} maintains only the consistency of the soft temporal plan without committing to a crisp solution. Only at the end of the planning process, a ground solution is found and scheduled for execution.

In other words, a crisp solution to a FTCN is a tuple of crisp values $s = (x_0, \ldots, x_{n+1})$ which represents an assignment of the form $X_i = x_i, x_i \in T$. In the framework of MACHINE^{TF}, a solution is an schedule of its actions, that is, an assignment of a crisp execution time for every action of the plan that may be used to execute the plan in practice. The set of solutions of a FTCN may be easily obtained by a solution extraction procedure (Fig. 11) and every solution has a degree of consistency which quantifies how accurate the solution is with respect to the constraints of the FTCN.

Definition 2 A σ-possible solution of a FTCN $\mathcal{N}$ is a tuple $s = (x_0, x_1, \ldots, x_{n+1})$ that verifies that $\pi_S(s) = \sigma$ and

$$\pi_S(s) = \min_{0 \le i,j \le n+1} \pi_{ij}(x_j - x_i). \tag{1}$$

Additionally, the most accurate solution that could be obtained from the set of solutions to a FTCN is given by the following value α, that may be used as a measure of the "goodness" of the FTCN (the fuzzy temporal plan).

Definition 3 A FTCN is α-consistent if the set of possible solutions $S \subseteq \mathcal{R}^{n+2}$ verifies

$$\sup_{s \in S} \pi_S(s) = \alpha. \tag{2}$$

Example 2 Table 1 shows a possible solution for the fuzzy plan of Fig. 9. This solution is 1-consistent, meaning that all the constraints have been completely satisfied.

Table 2 shows an alternative solution of the same plan but now the consistency is 0.73, meaning that some constraints have not been completely verified. In this case there is only one constraint that is not completely satisfied, and this one is the constraint between actions

Table 1 A consistent solution

OB	CB	REFUEL	STOP-REFUEL	ZOOM	STOP	OD	CD	END	σ
1	201	1	101	201	7201	7201	7401	7401	1

Table 2 A 0.73-consistent solution

OB	CB	REFUEL	STOP-REFUEL	ZOOM	STOP	OD	CD	END	σ
1	201	1	180	300	7340	7340	7526.6	7526.6	0.73

OD (Open Debarking) and CD (Close Debarking). This solution could have been obtained after the happening of some delay that make impossible the previous solution (in this case it is produced by a delay of action OD).

The equivalent to the classical (or not fuzzy) situation is the 1-consistency, or we can say only consistency. The inconsistency is related to the absence of solutions ($\alpha = 0$). When the network is 1-consistency then the distribution π_S is normalized, that is, there is at least a solution that completely verifies the desires of the user, although there may also be solutions with a different and intermediate consistency value.

In the framework of MACHINE^{TF}, the meaning of the value α is a generalization of the motivation for using fuzzy sets and it needs a further explanation. MACHINE^{TF} is deterministic, that is, all the plans found are valid to solve the problem, so the main difference between all possible valid plans is the degree of satisfaction of the soft temporal constraints α. Those valid plans with the highest value, $\alpha = 1$, are those plans such that there is at least a solution (schedule) that satisfy completely the desires of the user. On the contrary, those plans such that $0 < \alpha < 1$ are admissible plans, that is, plans that contain at least one admissible solution (schedule) that doesn't satisfy the original desires of the user but they still satisfy the constraints of the user at some degree greater than 0.

In this way, the consistency can be considered as a quality measure of the partial plan since reflects the desires of the user. In order to study the consistency of a network, next we present some previous concept and results described in (Marín et al. 1997).

Definition 4 Two FTCN $\mathcal{N}$ and $\mathcal{M}$ with the same number of variables are equivalent if and only if every σ-possible solution of one of them is also a σ-possible solution of the other, that is,

$$\pi_S^N(s) = \pi_S^M(s),$$

where π_S^N and π_S^M are the distribution associated with the fuzzy sets of the possible solutions of the FTCN $\mathcal{N}$ and $\mathcal{M}$ respectively.

An FTCN $\mathcal{M}$ is said to be a *minimal network* if its constraints are minimal, regarding inclusion, with respect to all its equivalent FTCNs $\mathcal{N}$. The constraints M_{ij} of the minimal network are obtained by means of an exhaustive propagation of constraints, through the following expression:

$$M_{ij} = \bigcap_{k=0}^{n+1} L_{ij}^k,$$

Fig. 10 Constraints propagation

```
for k=0 to n+1
  for i=0 to n+1
    for j=0 to n+1
      M_ij = M_ij ∩ (M_ik ⊕ M_kj)
  If M_ij = π_∅ then the network is inconsistent
```

where L_{ij}^k is the constraint induced by all the paths of length k that connect variables X_i and X_j:

$$L_{ij}^k = \bigcap C_{i_o,i_1,\ldots,i_k}^k, \quad i_1 \ldots i_{k-1} \leq n+1, i_0 = i, i_k = j,$$

$$C_{i_o,i_1,\ldots,i_k}^k = \sum_{p=1}^{k} L_{i_{p-1},i_p}.$$

The network $\mathcal{N}$ is inconsistent if and only if its minimal constraint is the empty distribution, that is $\pi_\emptyset(x) = 0\ \forall x \in \mathcal{R}$. The network is consistent (or 1-consistent) if and only if the constraints M_{ij} are normalized. The degree of consistency of the network is calculated by

$$\alpha = \sup_{s \in R^{n+2}} \pi_S(s) = \sup_{s \in R^{n+2}} \min_{0 \leq i,j \leq n+1} \pi_{ij}(x_j - x_i),$$

where each π_{ij} is associated the minimal constraint between the variables X_i and X_j.

The minimal network always verifies

$$M_{ij} \subseteq M_{ik} \oplus M_{kj}, \quad i, j, k \leq n+1.$$

This means that a new constraint propagation process would not provide any additional information on M_{ij}. In our case, the minimal network will be used to detect the consistency or inconsistency of the network. The algorithm to calculate the minimal network (see Fig. 10) is a fuzzy generalization (Marín et al. 1997; Vila and Godo 1994a) of the shortest-path algorithm proposed in (Dechter et al. 1991).

As the constraints have been defined as trapezoidal fuzzy numbers, we can easily manipulate them using the well known arithmetic described in (Dubois and Prade 1978). Thus, by using normalized trapezoidal fuzzy numbers this minimization algorithm is executed in polynomial time. The fuzzy operation required are:

$$(a, b, c, d) \oplus (e, f, g, h) = (a + e, b + f, c + g, d + h),$$

$$(a, b, c, d) \cap (e, f, g, h) = (\max\{a, e\}, \max\{b, f\}, \min\{c, g\}, \min\{d, h\}).$$

Finally, other important concept that we use in the interpretation of the use of the FTCN in the planning process is the independence between network variables.

Definition 5 Two variables X_i and X_j belonging to a FTCN $\mathcal{N}$ are independent if and only if the minimal constraints M_{ij}, M_{i0} and M_{0j} verify $M_{ij} = M_{i0} \oplus M_{0j}\ \forall i, j$.

When the variables are independent then the constraints M_{i0} and M_{j0} contains all the needed information to make variable assignment without consider the rest of constraints.

These results will be used in the next subsection to interpret and estimate the consistency of a network.

4.2 Consistency checking

The main goal of the scheduler module is determining the consistency of the network associated with the partial plan analyzed by the planner. In the fuzzy case, the consistency of the network is a degree value. Obviously, a consistency value $\alpha = 0$ implies the inconsistency of the network and there are no possible solutions. In other case, the scheduler returns to the planner a real value between 0 and 1. This value represents the degree in which the desires of the users are satisfied by the best solution of the network, and therefore it can be interpreted as a quality measure of the solutions associated with the network. The idea is to integrate this value in the heuristic function of the search process of the planner. Therefore, the first step is the calculus of the consistency of the network.

This calculus may seem easy but it poses a difficult question. Given (2), the value of the maximum degree of consistency α of a FTCN might be a very good source of information for decision making, but up to now, it cannot be obtained analytically.

In order to obtain the consistency value, we focus on the minimal network $\mathcal{M}$ of the original one. Thus, when all of the fuzzy temporal constraints in the minimal network are normalized, i.e.,

$$hgt(\pi_{ij}) = 1,$$

where

$$hgt(\pi) = \max_{t \in T} \pi(t),$$

the degree of consistency α is equal to 1, that is, there is at least one solution that completely meets the constraints (Marín et al. 1997) representing the initial desires of the user, otherwise $0 \leq \alpha < 1$ and thus, it is not possible to find a solution that verifies completely all the initial desires of the user but in any case verifies the soft constraints in a degree. In these cases some search algorithm like simulated annealing or genetic algorithms could be used to find that value, but this will degrade severely the performance of MACHINE^{TF} since this process should be launched to evaluate every possible plan being built.

In the framework of MACHINE^{TF}, although the value of α is not known, it may be bounded and used to guide the search of the planning process. Thus, the idea is to set bounds to the value of the consistency, and to use these bounds to estimate it.

The algorithm (Marín et al. 1997) described in Fig. 11 allow us to obtain solutions for a minimal FTCN once the consistency value is a known value.

In any case, we don't know the consistency value α, and therefore this procedure cannot be used directly. With the aim of obtaining a pessimistic estimation of the consistency we propose a greedy algorithm that consists in an iterative procedure that encapsulates the solution extraction algorithm.

Initially, we take $\sigma = 0$, and therefore we work on the support sets[1] of the fuzzy constraints of the minimal network. Following the algorithm of solution extraction we select a value of the support set for each one of the F sets. The set F contains in each step the possible values of each variable, taking into account the own constraints of the variable and the new constraints generated by the new assignments made by the algorithm. The selection procedure is not determined in the algorithm, therefore we have checked different possibilities (to select the minimum value of the support set, the center point of the support set, ...). The best results have been obtained when we select the minimum value of the mode, i.e., the core set of values of a fuzzy set $m(A) = \{u, \mu_A(u) = 1\}$ (Fig. 12).

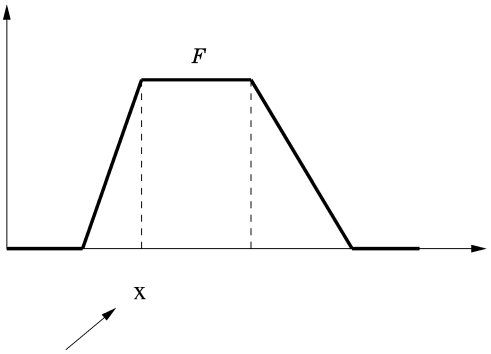
[1] The support of a fuzzy set A is the set $s(A) = \{u, \mu_A(u) > 0\}$.

Fig. 11 Solution extraction

Algorithm Solution Extraction
let $\langle X, M\rangle$ a minimal FTCN and $\sigma \in [0,1], \sigma < \alpha$

$X_0 = 0$
for i=1 to n+1
$F = \bigcap_{j<i} (x_j \oplus M_{ji})$
Select x_i such that $\pi_F(x_i) \geq \sigma$
$X_i = x_i$
$s = (x_0, x_1, \ldots, x_{n+1})$
Return s, $\pi_S(s) \geq \sigma$

Fig. 12 The minimum value of the mode of a fuzzy set



Once this value has been selected for each variable, we have a solution that can be evaluated through (1) obtaining a value σ_s. Now, we know that the value of the consistency of the network is equal or greater that σ_s since the consistency is the greatest π_S for all the possible solutions.

The next step consists in using again the algorithm of solution extraction but now with σ_s as new σ parameter. In this case we have selected a random value of the σ_s-cut of each F set. Following the same steps as above we obtain a new solution of the minimal network, and we repeat the process until we obtain $\sigma_s = 1$ or alternatively during a fixed number of runs. The infeasible solutions ($\sigma_s = 0$) are rejected, and the remaining ones are used to give a pessimistic estimation of the consistency:

$$\delta_l = \max_i \pi_S(\mathrm{sol}_i), \quad i = 1, \ldots, k,$$

where sol_i are the set of solutions considered by the previous process and k is the number of runs. Clearly,

$$\delta_l \leq \alpha.$$

On the other hand, let us consider the following value

$$\delta_u = \min_i hgt(\pi_{0i})$$

for a minimal FTCN. Now, we are going to prove that this value is an optimistic estimation of the consistency, that is, $\alpha \leq \delta_u$.

Proposition 1 *The consistency value α is always equal or less than the value* $\min_i hgt(\pi_{0i})$, *that is*, $\alpha \leq \delta_u$.

Proof Let $\delta_i = hgt(\pi_{0i})$ the height of the fuzzy number π_{0i}, since π_{0i} are trapezoidal fuzzy numbers

$$\exists x_i^* | \pi_{0i}(x_i^*) = \delta_i.$$

We can take as solution

$$s^* = (x_0^*, x_1^*, \ldots, x_{n+1}^*).$$

Because of the selection process of each x_i^*

$$\pi_{0i}(x_i) \leq \pi_{0i}(x_i^*)$$

for any x_i, then

$$\pi_S(s) = \min_{i,j} \pi_{ij}(x_j - x_i) \leq \min_i \pi_{0i}(x_i) \leq \min_i \pi_{0i}(x_i^*) = \min_i \delta_i = \delta_u$$

and therefore

$$\alpha = \sup_s \pi_S(s) \leq \delta_u. \qquad \square$$

Moreover, this result can be improved when the variables are independent. For each constraint M_{0i} we consider its modal interval defined by

$$L_i = \mathrm{mod}(M_{0i})$$

corresponding to all the elements of the domain whose membership function is equal to the height of the fuzzy constraints. We define

$$L = L_0 \times L_1 \times \cdots \times L_{n+1}.$$

When the variables are independent all the element of L are solutions of the network and the consistency coincide with the value of δ_u.

Proposition 2 *When the variables of the FTCN are all independent then every element s of L is a δ_u-possible solution of the network, and*

$$\alpha = \delta_u.$$

Proof For independent variable we know that

$$M_{ij} = M_{i0} \oplus M_{0j},$$

and therefore

$$M_{ij} = M_{0j} \ominus M_{0i}.$$

By taking a solution

$$s = (x_1, x_2, \ldots, x_{n+1}) \in L$$

then $x_i \in L_i$ y $x_j \in L_j$, and

$$(x_j - x_i) \in \text{mod}(M_{0j} \ominus M_{0i})$$

and

$$\pi_{ij}(x_j - x_i) = \min\{\delta_i, \delta_j\}$$

then

$$\min_{i,j} \pi_{ij}(x_j, x_i) = \min\left\{\min_i \pi_{0i}(x_i), \min_{i \neq 0, j} \pi_{ij}(x_j, x_i)\right\} = \delta_u$$

and obviously taking into account the selection of the elements of the solution

$$\alpha = \delta_u. \qquad \square$$

For independent variable the estimation of the consistency is really simple, since the modal interval of the constraints M_{0i} define completely the set of solutions. In the general case, the situation is more complex since the rest of binary constraints need to be taken into account, and the solution can be obtained outside the set L. In any case, the previous result gives us an interesting interpretation of the estimation δ_u. We can say that this is a rather good estimation since it has been obtained from a simplified model, that is, a model in which the variables have been considered as independent ones.

In any case, both estimation values δ_l and δ_u bounds the unknown value of α

$$\alpha \in [\delta_l, \delta_u].$$

Since the exact value of α is unknown, the centroid of this interval is used to approximate it. This value is then used by MACHINE^{TF} as a secondary ranking criterion during the search process, where the primary ranking criterion is the heuristic evaluation function of MACHINE, that has proven to be very useful in several domains (Castillo et al. 2001). This implies a deeper integration of planning and scheduling techniques since now the heuristic evaluation that guides the steps of the planner also takes into account the degree of temporal consistency in such a way that, given two plans with the same solving power, MACHINE^{TF} will choose that with the higher temporal consistency. Next section shows how the planner handles all the soft temporal constraints.

4.3 Handling soft temporal constraints during planning

The main loop of the planning algorithm is a best first search guided by a ranking function that takes into account the usefulness of the plan being constructed (Castillo et al. 2001) and the soft consistency of temporal constraints, as explained before. At every step, a pending subgoal is selected for its resolution so that either a new action or a existing one are used to solve it. In both cases new soft temporal constraints are posted to ensure a correct ordering of actions, propagated by means of an all-pairs polynomial algorithm (Fig. 10), and the soft temporal consistency of the resulting plan is evaluated and taken into account for its ranking. Eventually, if some interferences between concurrent actions is detected, this is corrected by adding new soft temporal constraints to produce a reordering of the actions that avoids the interference.

Goal satisfaction Let us suppose that an action a with an effect f that takes "about t_f time units", solves a precondition of an action b. Then the following soft temporal constraint

$$start(b) \geq start(a) \oplus (t_f - \delta, t_f, t_f, t_f + \delta)$$

is posted and propagated (Fig. 10). The satisfaction of a subgoal is recorded in a structure called causal link in order to avoid that no action that deletes f could overlap in the interval between $start(a)$ and $start(b)$.

Duration of actions For all actions, either with maximum duration $maxbound$ or not, we first calculate the following values:

$$dur^{\min}(a) = \max_{f' in\ effects(a)} (t_{f'} - \delta, t_{f'}, t_{f'}, t_{f'} + \delta),$$

$$dur^{\max}(a) = \begin{cases} (-\infty, -\infty, +\infty, +\infty) \text{ (unspecified)}, \\ (maxbound - \delta, maxbound, maxbound, maxbound + \delta) \text{ (bounded)} \end{cases}$$

and next we post and propagate the following constraint:

$$start(a) \oplus dur^{\max}(a) \geq end(a) \geq start(a) \oplus dur^{\min}(a).$$

Deadline goals For every deadline goal g, with deadline $(t_{\min} - \delta, t_{\min}, t_{\max}, t_{\max} + \delta)$, meaning that goal g must be achieved "more or less between times $t_{\min}$ and $t_{\max}$", which has been solved by an action a with its effect f that takes "about t_f time units", the following soft constraint is posted and propagated

$$start(a) = (t_{\min} - \delta, t_{\min}, t_{\max}, t_{\max} + \delta) \ominus (t_f - \delta, t_f, t_f, t_f + \delta).$$

Concurrent actions and threats Say that action c with the effect $(not\ f)$ that takes "about $t_{not(f)}$ time units" overlaps with a casual link from actions a to b with respect to the effect f. Then, the overlapping must be avoided by reordering the threatening action c by promotion (putting c after all the effects of b)

$$start(c) \geq start(b) \oplus dur^{\min}(b)$$

or demotion (putting a after the negative effect of c)

$$start(a) \geq start(c) \oplus (t_{not(f)} - \delta, t_{not(f)}, t_{not(f)}, t_{not(f)} + \delta).$$

With these new capabilities for handling soft temporal constraints, MACHINE^{TF} is able to obtain soft temporal plans like that shown in Fig. 9 by means of the solution extraction procedure shown in Fig. 11 and execute the plan. However, there are some additional features that must be mentioned.

5 Monitoring and rescheduling

One of the advantages of MACHINE^{TF} is that the underlying FTCN in a fuzzy temporal plan may be used to monitor its execution and to reschedule part of the plan in the case that a delay has occurred during its execution but maintaining the causal structure of the plan. In the case of MACHINE^{TF} there may be three different types of delays.

1. Initialization
   - Given $\Pi = \{a_0, a_1, \ldots, a_{n+1}\}$ a fuzzy temporal plan
   - $C = \emptyset$ and $Q = \emptyset$
   - Initialize $TIME$
2. Design a possible time line T
   - Fix the execution time for every action in C and consider $TIME$ to be the execution time for every action in Q
   - Use the solution extraction procedure to obtain a time line $s = \{x_i | a_i \in \Pi \cup Q\}$ with a degree of consistency $\sigma = \pi_S(s)$ given by equation 1 so that action a_i is scheduled to be executed at time x_i.
3. Monitoring the execution
   (a) If Π and Q are empty then **SUCCESS**
   (b) For every action $a_j \in \Pi$ such that $x_j = TIME$
      - Extract a_j from Π
      - If $preconds(a_j)$ are true in the environment, then execute a_j $(C = C \cup \{a_j\})$
      - Otherwise delay a_j $(Q = Q \cup \{a_j\})$
   (c) Advance $TIME$ in $\Delta TIME$
4. If $Q \neq \emptyset$ then monitor the queue.
   (a) For every action $a_j \in Q$
      - If $\pi_{0j}(TIME) = 0$ then **ERROR: TIME OUT**
      - If $preconds(a_j)$ are true in the environment, then extract a_j from Q and execute a_j $(C = C \cup \{a_j\})$
   (b) Go to Step 2
5. Go to Step 3

Fig. 13 Monitoring algorithm for fuzzy temporal plans

1. Local delays. They are delays that only affect locally to an isolated branch of the temporal plan without affecting the remaining actions. This is the easiest case since it could not affect deadline goals or makespans which depend on more global temporal constraints. In these cases, a new reschedule may be found only for the actions of the affected branch leaving the remaining schedule unaltered.
2. Global delays. They are more important since they may affect all of the remaining actions and deadline goals or makespans might also be affected and hence, a whole new reschedule might be needed.
3. Infeasible delays. When an action a_j has been delayed more than it is acceptable even taking into account the existence of soft temporal constraints, i.e., $\pi_{0j}(delay) = 0$, then no reschedule is possible and the possibility of re-planning should be considered.

The algorithm to monitor the execution of fuzzy temporal plans is shown in Fig. 13. It uses the set of actions of the plan still to be executed Π, a set of already executed actions C and a queue of delayed actions Q, that is, actions that should have executed before but they are not able to execute because, for some reason, some of its preconditions have not been achieved yet by their producing actions. The variable *TIME* is used to track the evolution of the schedule. It works as follows. The monitoring procedure defines a tentative schedule by obtaining a solution with maximal consistency. If nothing goes wrong ($Q = \emptyset$), every

action whose execution time equals to *TIME* and whose preconditions have already been satisfied, is executed and the variable *TIME* is increased to the earliest execution time of the remaining actions in Π.

However a delay (either with local scope or global scope) might occur at time *TIME*, that is, some of the effects of an action might take longer than expected producing the delay of all of the actions that had been scheduled to time *TIME* but that have that missing effect in their preconditions. In this case, delayed actions are included in the queue Q and variable *TIME* is continuously increased by a minimum $\Delta TIME$ producing, in every iteration, a recomputation of the schedule for actions either in Q or Π. This reschedule of the remaining actions in the plan is needed to propagate the delay of the actions in Q to any future action that could have any temporal constraint relative to them. This procedure allows to readapt the schedule to the detected delay. It must be said that, in the case of delays, since new schedules are being continuously obtained to fit the delay, the consistency σ of the schedules may decrease due to the violation of any fuzzy temporal constraint but it will still be acceptable whenever $\sigma > 0$. The extreme case is when the accumulated delay is completely unacceptable and the consistency of the schedule is 0. This is detected in step 4 when the variable *TIME* takes an infeasible value for some action a_j such that $\pi_{0j}(TIME) = 0$.

This capability for monitoring fuzzy temporal plans is very useful in realistic domains. One could have argued that this could have also been achieved by obtaining a rigid temporal plan and executing it in a flexible manner, however this would pose severe questions on the consistency of the explicit delay of an action since it might produce unsolvable flaws with respect to future actions causally dependent of the delayed action or produce any unexpected interference with the effects of other concurrent actions. In the case of MACHINE^{TF}, the delay of actions in order to flexibly modify a previous schedule is a safe process since it is based on the fuzzy temporal constraints explicitly included in the plan, which have been obtained taking into account the existence of threats and causal relations between actions, as explained in previous sections. Hence, no feasible delay nor reschedule obtained on the basis of the FTCN of the plan could produce an unexpected interference between parallel actions, either on local delays or global delays.

6 An example

The following example shows the monitoring of the plan in Fig. 9 and simulates the occurrence of several unexpected delays as well as the corresponding reschedule of actions. The simulation is shown in Table 3.

The first row is the initial schedule with $\sigma = 1$ and it is shown in Fig. 14a. Once the execution starts, four delays have been simulated.

1. The first one (`Stop-Refuel`) is a local delay that only affects one action. The remaining delays are global ones.
2. It may also be seen that the first global delay (`ZOOM`) only produces a reschedule of the actions without affecting the consistency (quality) of the plan ($\sigma = 1$, Fig. 14b).
3. However the second global delay (`STOP`) degrade a little the consistency of the plan, that is, it produces a delay that does not completely satisfy the strict makespan, but it might be acceptable in the terms of the fuzzy temporal constraints represented in the domain ($\sigma = 0.73$). It is worth noting that the solution extraction procedure also suggest reducing the expected duration of action `Close-Debark` (`CD`) in order to maintain the highest consistency. The duration of action `CD` encoded in the domain is the fuzzy subset $\pi_{OD-CD} \equiv (150\ 200\ 300\ 350)$, its duration in every previous schedule had been fixed to

Table 3 An example of the execution of the plan in Fig. 9 which simulates several delays and their corresponding reschedules. Times in [brackets] represent already executed actions and times in (parentheses) represent delayed actions

TIME	OB	CB	REFUEL	STOP-REFUEL	ZOOM	STOP	OD	CD	END	Detected Delay	σ
0	1	201	1	101	201	7201	7201	7401	7401	–	1
100	[1]	201	[1]	101	201	7201	7201	7401	7401	–	1
101	[1]	201	[1]	(101)	201	7201	7201	7401	7401	STOP-REFUEL	1
180	[1]	201	[1]	[180]	201	7201	7201	7401	7401	–	1
201	[1]	[201]	[1]	[180]	(201)	7201	7201	7401	7401	ZOOM	1
300	[1]	[201]	[1]	[180]	[300]	7300	7300	7500	7500	–	1
7300	[1]	[201]	[1]	[180]	[300]	(7300)	7300	7500	7500	STOP	1
7340	[1]	[201]	[1]	[180]	[300]	[7340]	[7340]	7526.6	7526.6	–	0.73
7526.6	[1]	[201]	[1]	[180]	[300]	[7340]	[7340]	(7526.6)	7526.6	CD	0.73
7540	[1]	[201]	[1]	[180]	[300]	[7340]	[7340]	[7540]	[7540]	–	0.6

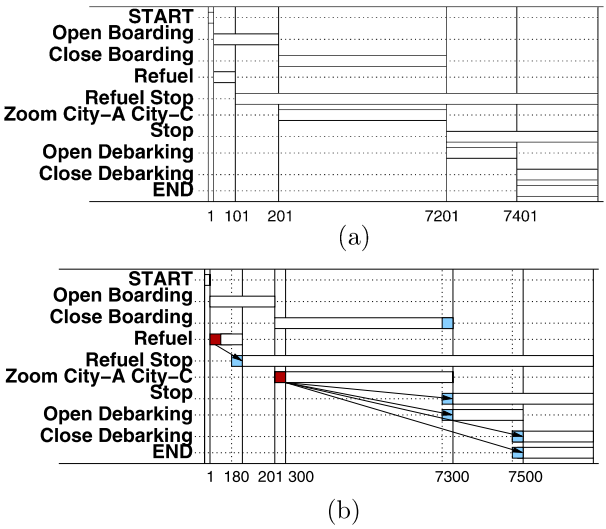


Fig. 14 Two different schedules obtained from the plan in Fig. 9. **a** The original schedule with $\sigma = 1$. **b** The reschedule after the second delay with $\sigma = 1$

200 time units ($\pi_{OD-CD}(200) = 1$) but after this delay, the solution extraction procedure suggests to fix it to 186.6 time units ($\pi_{OD-CD}(186.6) = 0.73$).

4. However, this is not finally possible and there is a new delay of action CD that ends the plan with a consistency of $\sigma = 0.6$.

7 Some experiments

This section shows the performance of MACHINE^{TF} in several realistic problems and one problem based on the zeno travel domain shown along the paper. The description of these problems is as follows.

Problem # 1. It is the problem in the zeno travel domain, as seen up to now with fuzzy durations.

Problem # 2. It is a sample problem in the fire fighting scenario.

Problem # 3. MACHINE^{TF} is also able to deal with problems from other domains, like manufacturing systems. This third problem is a real-life example of batch manufacturing for a dairy products problem (Castillo et al. 2000, 2001) with fuzzy due dates. It consists of 48 different possible actions.

The results of the execution of MACHINE^{TF} in these domains are shown in Table 4. In realistic domains with a complex knowledge representation, run times do not seem to scale as efficiently as many of the high performance planners that take part in the International Planning Competition (Long and Fox 2003a) but, on the other hand, its temporal expressiveness enable it to deal with realistic problems where these other planners may not succeed, that is, problems with ill defined temporal knowledge, to obtain approximately consistent temporal plans and to adapt the execution of these plans to unexpected delays during the execution.

Table 4 Experimental results of MACHINE^{TF} in several domains. (DS) Domain Size: number of possible actions. (PL) Plan length: number of actions included in the plan. (NE) Search effort: nodes of the search space explored by the planning algorithm. (CT) CPU time in seconds, running CLISP and Linux on a Pentium IV 1.6 GHz

Problem	DS	PL	NE	CT
P#1	9	18	28	2.1
P#2	17	19	337	27
P#3	48	40	1913	1619.7

8 Final remarks

This paper has outlined MACHINE^{TF}, a temporal planner based on the representation of fuzzy temporal knowledge able to build fuzzy temporal plans. The main contributions of MACHINE^{TF} are the following.

- It is able to obtain plans in domains in which time is not precisely known. This is very important in real world problems and mainly with respect to classic temporal planners and schedulers, which need a rigid representation of time and, therefore, they lose much of the temporal expressiveness of these domains.
- It is also able to obtain approximate solutions where exact solutions are not feasible, given a soft interpretation of temporal constraints. This is also very important in hard temporal problems since it transforms the inconsistency in a matter of preference and degrees, more in accordance with most usual constraints in real life.
- And finally, fuzzy temporal plans obtained provide a flexible time line for its execution. Given that a fuzzy temporal plan may be scheduled in different time lines, depending on the solution that has been extracted, if an unexpected delay occurs (due to an error, a delay in the execution of an action, etc.), a new schedule adapted to the new situation may be re-extracted without the need to re-plan.

To illustrate the results of these techniques, some examples extracted from the domain of crisis management have been shown. These examples are inspired in the work being done by the authors under a research contract with the Andalusian Regional Ministry of Environment (de la Asunción et al. 2003, 2005) devoted to an intelligent decision support system for the assisted design of forest fire fighting plans.

However there are some open problems that need further work. The first one is the need to improve the featuring of FTCNs in order to obtain a better estimation of the global consistency α. The second one is related to the monitoring procedure. As explained before, in the case of delays ($Q \neq \emptyset$) the procedure continuously reschedules the remaining actions propagating the delay along the existing constraints. This is mainly done by a shortest path algorithm whose efficiency is $O(n^3)$ where n is the number of time points (actions) of the FTCN. This means that, if the number of actions is very large and the needed temporal resolution to advance *TIME* is very small, there might be no time to complete the propagation. Therefore, a post-processing of the FTCN that could restrict the propagation of constraints to make the reschedule faster might be used. In fact, a reformulation like the one in proposed in (Morris and Muscettola 2000) defined on TCNs could be adapted to the case of fuzzy constraints to cope with degrees of consistency and applied to the fuzzy temporal plans of MACHINE^{TF} before the monitoring procedure.

9 Related work

MACHINE^{TF} is a deterministic planner and uncertainty is restricted only to the time of occurrence of events so it is a classical goal-oriented planner leaving out of its scope other approaches to handle uncertainty that involve nondeterminism and a planning process focused on optimizing a plan utility function like in MDPs (Blythe 1999; Bresina et al. 2002). However it still faces a very important problem in deterministic domains, that is, how to obtain and execute a plan when part (or the whole) of the temporal knowledge is uncertain.

The need to represent and reason about uncertain temporal knowledge has been a very active field in the literature devoted to real problems. The approach presented in this paper is based on the use of possibility distributions (Zadeh 1978) to model either uncertainty or imprecision or a preference criterion in same the sense of (Khatib et al. 2001) (in fact fuzzy temporal constraints are a special case of this approach) and not necessarily related to probabilistic uncertainty like in (Bresina et al. 2002). There are also other approaches that deal with temporal uncertainty that have been built over STN (Dechter et al. 1991) like STNUs, simple temporal networks under uncertainty (Vidal and Fargier 1999; Morris and Muscettola 2000). These approaches are based on the use of contingent (uncertain) constraints defined on pairs of temporal points to represent uncertain temporal knowledge. Contingent temporal constraints do no explicitly affect the consistency of the STNU, that is still a crisp result in $\{0, 1\}$, although it does affect the schedule of a solution. The duration of a contingent constraint cannot be predicted before execution, and variables related to a contingent constraint must be maintained uninstantiated until the execution of the plan.[2] In the case of MACHINE^{TF}, fuzzy temporal constraints provide a little more of expressiveness and they define a possibility measure over these contingent links to represent degrees of uncertainty or preference so final solutions will also have associated a σ-consistency varying softly in $[0, 1]$. Although MACHINE^{TF} can make predictions, with maximal possibility, on these contingent links during the monitoring of the plan, there is no way to guarantee that this prediction will be finally obtained in a σ-consistent real schedule nor that the final schedule will be σ-consistent, $\sigma > 0$.

There are also works on scheduling temporal plans either on STNs (Muscettola et al. 1998) or STNUs (Morris and Muscettola 2000) but in all of them any schedule may suddenly change from consistency to inconsistency. In MACHINE^{TF} there are degrees of consistency allowing for approximate plans or schedules when completely consistent ones are not possible.

References

Avesani, P., Perini, A., & Ricci, F. (2000). Interactive case-based planning for forest fire management. *Applied Intelligence*, *13*(1), 41–57.

Bacchus, F., & Ady, M. (2001). Planning with resources and concurrency. A forward chaining approach. In *IJCAI'01*.

Bartõk, R., & Mecl, R. (2003). Integrating planning into production scheduling: Visopt shopfloor system. In G. Kendall, E. Burke & S. Petrovic (Eds.), *Proceedings of the 1st multidisciplinary international conference on scheduling: theory and applications (MISTA)* (pp. 259–278).

Bienkowski, M. (1995). Demonstrating the operational feasibility of new technologies: the ARPI IFDs. *IEEE Expert*, *10*(1), 27–33.

[2]This leads to the definition of several types of controllabilities (strong, dynamic and weak) depending on how one may ensure, before execution, that a consistent solution may be finally scheduled.

Biundo, S., & Schattenberg, B. (2001). From abstract crisis to concrete relief—a preliminary report on combining state abstraction and htn planning. In *6th European conference on planning (ECP-01)*.

Biundo, S., Aylett, R., Beetz, M., Borrajo, D., Cesta, A., Grant, T., McCluskey, L., Milani, A., & Verfaillie, G. (2003). *PLANET technological roadmap on AI planning and scheduling*. Electronically available at http://www.planet-noe.org/service/Resources/Roadmap/Roadmap2.pdf.

Blythe, J. (1999). Decision-theoretic planning. *AI Magazine*, *20*(2), 37–54.

Bresina, J., Dearden, R., Meuleanu, N., Ramakrishnan, A., Smith, D., & Washington, R. (2002). Planning under continuous time and resource uncertainty: a challenge for AI. In *Proc. UAI*.

Castillo, L., Fdez-Olivares, J., & González, A. (2000). Automatic generation of control sequences for manufacturing systems based on nonlinear planning techniques. *Artificial Intelligence in Engineering*, *4*(1), 15–30.

Castillo, L., Fdez-Olivares, J., & González, A. (2001). Mixing expresiveness and efficiency in a manufacturing planner. *Journal of Experimental and Theoretical Artificial Intelligence*, *13*, 141–162.

Cohen, P., Greenberg, M., Hart, D., & Howe, A. (1989). Trial by fire: understanding the design requirements for agents in complex environments. *AI Magazine*, *10*(3), 32–48.

de la Asunción, M., Castillo, L., Fdez-Olivares, J., García-Pérez, O., González, A., & Palao, F. (2003). SIADEX: assisted design of forest fire fighting plans by artificial intelligence planning techniques. http://siadex.ugr.es.

de la Asunción, M., Castillo, L., Fdez-Olivares, J., García-Pérez, O., González, A., & Palao, F. (2005). SIADEX: an interactive artificial intelligence planner for decision support and training in forest fire fighting. *Artificial Intelligence Communications*, 18(4).

Dechter, R. (2003). *Constraint processing*. Morgan Kaufmann.

Dechter, R., Meiri, I., & Pearl, J. (1991). Temporal constraint networks. *Artificial Intelligence*, *49*, 61–95.

Do, M., & Kambhampati, S. (2001). SAPA: a domain-independent heuristic metric temporal planner. In *European conference on planning* (pp. 109–120).

Dubois, D., & Prade, H. (1978). Operations on fuzzy numbers. *International Journal of Systems Science*, *9*, 613–626.

Dubois, D., Fargier, H., & Prade, H. (1993). The use of fuzzy constraints in job-shop scheduling. In *Proc. of IJCAI-93/SIGMAN workshop on knowledge-based production planning, scheduling and control*, Chambery, France.

Dubois, D., Fargier, H., & Fortemps, P. (2003). Fuzzy scheduling: modelling flexible constraints vs. coping with incomplete knowledge. *European Journal of Operational Research*, *147*, 231–252.

Haslum, P., & Geffner, H. (2001). Heuristic planning with time and resources. In *European conference on planning* (pp. 121–132).

Khatib, L., Morris, P., Morris, R., & Rossi, F. (2001). Temporal constraint reasoning with preferences. In *IJCAI 2001* (pp. 322–327).

Laborie, P., & Ghallab, M. (1995). Planning with sharable resource constraints. In *IJCAI'95* (pp. 1643–1649).

Long, D., & Fox, M. (2003a). The 3rd international planning competition: results and analysis. *Journal of Artificial Intelligence Research*, *20*, 1–59.

Long, D., & Fox, M. (2003b). PDDL2.1: an extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research*, *20*, 61–124.

Marín, R., Cárdenas, M., Balsa, M., & Sánchez, J. (1997). Obtaining solutions in fuzzy constraint networks. *International Journal of Approximate Reasoning*, *16*, 261–288.

Morris, P., & Muscettola, N. (2000). Execution of temporal plans with uncertainty. In *AAAI 2000* (pp. 491–496).

Munoz-Avila, H., Aha, D. W., Breslow, L., & Nau, D. (1999). HICAP: an interactive case-based planning architecture and its application to noncombatant evacuation operations. In *Ninth conference on innovative applications of artificial intelligence* (pp. 879–885). AAAI Press.

Muscettola, N. (1994). HSTS: integrating planning and scheduling. In M. Zweben & M. Fox (Eds.), *Intelligent scheduling* (pp. 169–212). Morgan Kaufmann.

Muscettola, N., Morris, P., & Tsamardinos, I. (1998). Reformulating temporal plans for efficient execution. In *6th conf. on principles of knowledge representation and reasoning* (pp. 444–452).

Myers, K. L. (1999). CPEF: A continuous planning and execution framework. *AI Magazine*, *20*(4), 63–69.

Penberthy, J., & Weld, D. (1994). Temporal planning with continous change. In *AAAI'94* (pp. 1010–1015).

Smith, D., & Weld, D. (1999). Temporal planning with mutual exclusion reasoning. In *IJCAI'99* (pp. 326–337).

Vidal, T., & Fargier, H. (1999). Handling contingency in temporal constraint networks: from consistency to controllabilities. *Journal of Experimental and Theoretical Artificial Intelligence*, *11*, 23–45.

Vila, L., & Godo, L. (1994a). On fuzzy temporal constraint networks. *Mathware and Soft Computing*, *3*, 315–334.

Vila, L., & Godo, L. (1994b). Query answering in fuzzy temporal constraint networks. In *IJCAI'95: Proceedings of the international joint conference on artificial intelligence* (Vol. 3, pp. 315–334).
Weld, D. (1994). An introduction to least commitment planning. *AI Magazine*, *15*(4), 27–61.
Wilkins, D. E., & Desimone, R. V. (1994). Applying an AI plannet to military operations planning. In M. Zweben & M. S. Fox (Eds.), *Intelligent scheduling*. Morgan Kaufmann.
Zadeh, L. (1978). Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, *1*, 3–28.