Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

Strong Temporal Planning with Uncontrollable Durations: A State-Space Approach

Alessandro Cimatti and **Andrea Micheli** and **Marco Roveri**
Fondazione Bruno Kessler – Italy
{cimatti,amicheli,roveri}@fbk.eu

Abstract

In many practical domains, planning systems are required to reason about durative actions. A common assumption in the literature is that the executor is allowed to decide the duration of each action. However, this assumption may be too restrictive for applications.

In this paper, we tackle the problem of temporal planning with uncontrollable action durations. We show how to generate robust plans, that guarantee goal achievement despite the uncontrollability of the actual duration of the actions. We extend the state-space temporal planning framework, integrating recent techniques for solving temporal problems under uncertainty. We discuss different ways of lifting the total order plans generated by the heuristic search to partial order plans, showing (in)completeness results for each of them.

We implemented our approach on top of COLIN, a state-of-the-art planner. An experimental evaluation over several benchmark problems shows the practical feasibility of the proposed approach.

Introduction

Many planning domains (e.g. aerospace, logistics) require the ability to reason about durative actions. Over the years, the problem has received significant attention, and several interesting approaches, implemented in various practical planners, have been proposed (Penberthy and Weld 1994; Younes and Simmons 2003; Coles et al. 2009; 2012a). Common assumptions are that either the duration of each action is fixed in the domain or the plan executor is allowed to specify the start time as well as the duration of the actions. In many practical applications, however, the duration of actions cannot be controlled. As an example, consider a navigation task for an exploratory rover: the duration may vary depending on unknown operational conditions (e.g. soil, available power), and can at best be assumed to lie within a given range. In order to properly reason in these situations, a planner must be able to take this uncontrollability into account.

In this paper, we tackle the problem of generating *strong time-triggered plans*, i.e. plans where the start time of actions is predetermined, and that guarantee goal achievement for any admissible value of uncontrollable durations. These plans are similar to conformant plans in non-deterministic planning, but here we allow for uncertainty only in the action duration and we consider a dense model of time. Similarly to conformant planning, this problem is important in practice, even if many instances do not admit a solution. We remark that synthesizing plans that are *guaranteed* to work under any possible contingency, is valuable in contexts where safety must be certified.

We make the following contributions. First, we formally define a framework to express the uncontrollability in the action durations: we define the strong planning problem with temporal uncertainty, we present strong time triggered plans (STTP) as solutions to the defined planning problem, and we formally define the notion of validity for a given STTP.

Second, we extend forward state-space temporal planning (FSSTP) (Coles et al. 2009; 2012a) to solve strong planning problems with temporal uncertainty. The FSSTP framework relies on the interplay between a discrete planner to generate a totally ordered plan for an abstraction of the domain, and a temporal reasoner to check the temporal consistency of the corresponding Temporal Problem (TP). We retain the generation of abstract plans, and, to deal with uncontrollability of action duration, we replace the solver for temporal problems with a solver for Temporal Problems with Uncertainty (TPU) (Vidal and Fargier 1999). We consider three different ways of creating the temporal problems by lifting the totally-ordered abstract plans. The first two cases, are both derived from existing approaches in temporal planning: they construct a temporal problem relying on a total order (Coles et al. 2012a) and on a partial order (Coles et al. 2010) among actions. Both approaches, however, are shown to be incomplete in presence of temporal uncertainty. The third case, called Disjunctive Reordering (DR), achieves completeness by generating a Disjunctive Temporal Problem with Uncertainty (DTPU) (Peintner, Venable, and Yorke-Smith 2007) that takes into account all possible sets of valid reorderings of the given abstract plan.

Finally, we implemented these approaches in a planner, built extending COLIN (Coles et al. 2012a). Our planner is able to process an extension of the PDDL 2.1 language that allows for expressing domains having actions with uncontrollable duration. We carried out an experimental evaluation on several benchmarks, both new and adapted from the 2011 international planning competition (Coles et al. 2012b).

Related Work. Temporal planning is a vast and diverse area of research. Solving a temporal planning problem means to guarantee that the solution plans are causally sound (they lead to the goal, satisfying action applicability) and that the timing constraints are satisfied. Many models and techniques have been proposed to deal with this problem (Penberthy and Weld 1994; Frank and Jónsson 2003; Younes and Simmons 2003; Coles et al. 2009; 2012a).

To the best of our knowledge, no previous work addresses uncontrollable durations in the setting of action-based temporal planning. The only work tackling planning with uncontrollable durations is our previous work (Cimatti, Micheli, and Roveri 2013) in the setting of timeline-based planning (Frank and Jónsson 2003; Cesta et al. 2009). There are profound differences between action-based and timeline-based temporal planning, but discussing them is out of the scope of this paper. Here suffices to say that in timeline-based planning (with controllable durations), the domain description is not action-based, but rather represented by means of a set of rules on the evolution of "state variables", subject to complex synchronization constraints. The presented approach constructs strong time triggered plans similarly to what we propose in this paper. However, that work is mainly theoretical and the proposed reduction to (decidable) first order logic requires to fix a maximum time horizon.

In this paper, we explicitly model temporal uncertainty in the planning domain and we synthesize plans that are guaranteed to achieve the goal despite such uncertainty. A different approach to deal with uncontrollable durations is based on the idea of "flexible execution" (Cesta et al. 2009; Muise, Beck, and McIlraith 2013). Instead of modeling uncertainty in the domain, the planner identifies a suitable set of actions to achieve the goal, together with a (compactly represented) set of dependencies (e.g. an STP). The plan executor is then equipped with an online temporal reasoner to schedule the actions according to observations. We remark, that this approach does not guarantee goal achievement under all uncontrollable durations, as uncertainty is not modeled, but tries to grant some freedom for the executor of the plan. We believe that controllability and flexibility are not mutually exclusive one another: it is possible to conceive flexible controllable plans (Wilson et al. 2014). This is however out of the scope of this paper and left for future work.

Background

Temporal reasoning. In temporal planning, we need to reason not only on the ordering of actions in time, but also on their metric duration. A common framework to represent and reason on this kind of constraints is the Temporal Problem (TP) (Dechter, Meiri, and Pearl 1991; Tsamardinos and Pollack 2003). A TP is a formalism used to represent temporal constraints over time-valued variables representing time points. Each constraint is a disjunction of atoms in the form $x - y \in [l, u]$, where x and y are time points and $l, u \in \mathbb{R}^+$. Formally, a TP is a tuple (X, C) where X is a set of time points and C is a set of temporal constraints. Each time point is typically used to represent the start or the end of an action; causal relations are represented as precedence constraints, forcing an event to happen before another. Finally, metric constraints allow for the encoding of action durations. A solution to a TP is an assignment of real values to the time points that fulfills all the constraints. A TP is said to be consistent if it has a solution.

In order to deal with temporal uncontrollability, TP with uncertainty (TPU) have been proposed (Vidal and Fargier 1999; Peintner, Venable, and Yorke-Smith 2007).

Definition 1. *A TPU is a tuple (X_c, X_u, C_c, C_f), where X_c and X_u are sets of time points and C_c and C_f are sets of temporal constraints.*

In a TPU some of the time points can be assigned by the solver (X_c), while the others (X_u) are intended to be assigned by the environment. Similarly, the constraints are divided in requirements (called free constraints, C_f) and assumptions (called contingent constraints, C_c). As such, TPUs can be seen as a form of game between the solver and an adversarial Nature (Cimatti, Micheli, and Roveri 2014).

Different kinds of queries can be issued given a TPU. In this paper, we focus on strong controllability (SC) (Vidal and Fargier 1999; Peintner, Venable, and Yorke-Smith 2007). A TPU is strongly controllable if there exists an assignment (called *strong schedule*) of real values to each controllable time point, such that all free constraints are satisfied for every possible assignment of the uncontrollable time points fulfilling the contingent constraints.

Depending on the structure of the constraints, various classes of TPUs have been identified (Peintner, Venable, and Yorke-Smith 2007). We focus on two classes of TPUs: Simple Temporal Problems with Uncertainty (STPU) and Disjunctive Temporal Problems with Uncertainty (DTPU). An STPU is a TPU where each constraint has exactly one disjunct (i.e. it is conjunctive), while DTPUs allow for arbitrary Boolean combinations in the constraints.

Several approaches to check SC of a TPU have been proposed. Vidal and Fargier (1999) show that the strong controllability problem for an STPU is polynomial-time, while for DTPU it is NP-hard (Peintner, Venable, and Yorke-Smith 2007). Recently, new techniques to solve SC for DTPU have been presented (Cimatti, Micheli, and Roveri 2014): they rely on the reduction of SC to a Satisfiability Modulo Theory (Barrett et al. 2009) problem.

Temporal planning. Following Ghallab, Nau, and Traverso (2004), a (STRIPS) classical planning problem is a tuple (F, O, I, G), where F is the set of fluents, O is a set of actions, $I \subseteq F$ is the initial state, and $G \subseteq F$ is a goal condition. A literal is either a fluent or its negation. Every action $a \in O$ is defined by two set of literals: the preconditions, written $pre(a)$, and the effects, $eff(a)$. A classical plan $\chi = (a_1, \cdots, a_n)$ is a sequence of actions. A plan is valid if and only if it is executable from the initial state and terminates in a state fulfilling G.

Similarly to Cushing et al. (2007), we define a temporal planning problem as a tuple (F, DA, I, G), with F, I and G as before and with DA being a set of durative actions. A durative action A is given by two classical planning actions: $begin(A)$, and $end(A)$, an overall condition $overall(A)$ expressed as a set of literals, and a minimum $\delta_{min}(A) \in \mathbb{R}^+$ and maximum $\delta_{max}(A) \in \mathbb{R}^+$ duration (with

Algorithm 1 The FSSTP framework

```
1: procedure FSSTP(P)
2:     for all partial χ generated while solving abs(P) do
3:         D ← DURATIONS(χ, P)
4:         P ← PRECEDENCES(χ, P)
5:         if μ ← TP.SOLVE((χ, D ∪ P)) then
6:             if ISCOMPLETE(χ) then
7:                 return BUILDTEMPORALPLAN(μ, χ)
8:             else CONTINUE( )
9:         else REJECT(χ)
10:    return ⊥
```

$\delta_{min}(A) \leq \delta_{max}(A)$). A temporal plan $\pi = \{s_1, \cdots, s_n\}$ is a set of steps, where each step s is composed of a starting time $t(s) \in \mathbb{R}^+$, an action $action(s)$ and a duration $\delta(s) \in [\delta_{min}(action(s)), \delta_{max}(action(s))]$. Due to space constraints we do not give the precise semantics of a valid temporal plan: see Fox and Long (2003) for a thorough description. For this paper we say that a *temporal plan is valid* if and only if it can be simulated: starting from the initial model we apply each step at time $t(s)$ and at the end of the simulation we obtain a state fulfilling the goal condition.

We focus on a specific approach to temporal planning among others: forward state-space temporal planning (FSSTP). The idea of this approach is to create an interplay between a state-based forward search planner to generate a plan sketch and a temporal reasoner to check its temporal feasibility (Coles et al. 2008; 2012a). Each durative action A is expanded in a pair of classical planning actions called *snap actions*: $A_\vdash$ corresponding to $begin(A)$, and $A_\dashv$ corresponding to $end(A)$. $A_\vdash$ and $A_\dashv$ have the preconditions and the effects of their corresponding classical planning actions, and the overall condition of A as additional precondition. We force the planner to instantiate snap actions in pairs (each start is coupled with exactly one end) and forbid any action threatening the overall condition of A between the snap actions for A (Coles et al. 2009).

Definition 2. *Given a temporal planning $\mathcal{P} = (F, DA, I, G)$, the **abstraction of** $\mathcal{P}$ (written $abs(\mathcal{P})$) is a classical planning problem $(F, \bigcup_{A \in DA}\{A_\vdash, A_\dashv\}, I, G)$.*

The pseudo-code of a forward state-space temporal planner (FSSTP) is shown in Algorithm 1. A classical planner implemented as a forward state-space search solves the abstract problem $abs(\mathcal{P})$. The planner keeps a totally-ordered partial plan χ. Each time an action is added to the partial plan, the scheduling check is invoked to assess the temporal consistency of the added action. The scheduling check builds a TP[1] that has the steps of χ as time points and has a set of constraints composed of duration constraints D and precedence constraints P. Duration constraints (created by the DURATIONS function) are used to bind pairs of snap actions $(A_\vdash, A_\dashv)$, forcing the duration of each action to obey the domain specification $(A_\dashv - A_\vdash \in [\delta_{min}(A), \delta_{max}(A)])$. Precedence constraints (created by the PRECEDENCES function) are used to maintain causality in the plan. If a step a is needed to achieve a precondition for another step b, we must impose a precedence among the two steps ($a < b$). Similarly, precedence constraints are used to impose that the $overall(A)$ conditions for an action A are maintained.

When the TP is found to be consistent, two situations can occur. If χ was a plan achieving the goal in $abs(\mathcal{P})$, we can terminate the procedure, otherwise we continue the search in the abstract domain (CONTINUE). To terminate, we build a temporal plan π from a consistent schedule μ of the TP[2]: each pair of snap actions $A_\vdash$, $A_\dashv$ in χ is a step in π, the time for the step is $\mu(A_\vdash)$ and the duration is $\mu(A_\dashv) - \mu(A_\vdash)$. If the TP is not consistent, the classical planner backtracks, as χ is not temporally sound and cannot be further extended.

Strong Planning With Temporal Uncertainty

In this section, we define the semantics of the *strong planning problem with temporal uncertainty* and we show the issues that arise in comparison to temporal planning. The only syntactic modification we require in the temporal planning problem is to have a partition of the durative actions DA in controllable (DA_c) and uncontrollable (DA_u). Intuitively, a controllable durative action behaves as in plain temporal planning: the planner is free to assign both the starting time and the duration of the action. Instead, an uncontrollable durative action ($A \in DA_u$) is started by the planner, but the duration is not controllable: we only assume that the duration will be in the $[\delta_{min}(A), \delta_{max}(A)]$ interval.

Definition 3. *A **strong planning problem with temporal uncertainty** $\mathcal{U} = (F, DA_c, DA_u, I, G)$ is a tuple where DA_c and DA_u are two disjoint sets of durative actions, and $(F, DA_c \cup DA_u, I, G)$ is a temporal planning problem.*

We call *Strong Time Triggered Plan* (STTP) a solution to a strong planning problem with temporal uncertainty.

Definition 4. *An **STTP** σ is a set of steps $\{r_1, \cdots, r_n\}$ where each step r is composed of a starting time $t(r) \in \mathbb{R}^+$, an action $action(r)$ and, if $action(r) \in DA_c$, a duration $\delta(r) \in [\delta_{min}(action(r)), \delta_{max}(action(r))]$.*

Syntactically, an STTP is similar to a temporal plan, but we do not assign the duration of uncontrollable actions. Semantically, an STTP represents a set of temporal plans, one for each possible duration of each uncontrollable action.

Definition 5. *An STTP $\sigma = \{r_1, \cdots, r_n\}$ **induces a plan** $\pi = \{s_1, \cdots, s_n\}$ if for each i, $action(s_i) = action(r_i)$, $t(s_i) = t(r_i)$ and the following conditions hold.*

1. *$\delta(s_i) = \delta(r_i)$, if $action(r_i) \in DA_c$*
2. *$\delta(s_i) \in [\delta_{min}(action(r_i)), \delta_{max}(action(r_i))]$, otherwise.*

Given an STTP σ, we write $\mathcal{I}_\sigma$ to indicate the set of all induced plans $\{\pi | \pi \text{ is induced by } \sigma\}$. Clearly, the set of induced plans for a strong planning problem with temporal uncertainty is usually infinite.[3] This definition, however, allows for the formalization of validity for an STTP exploiting

[1] We consider purely temporal planning, other works cope with numeric fluents and continuous effects by using linear programs instead of temporal problems (Coles et al. 2012a; 2010).

[2] We write $\mu(x)$ to indicate the value assigned to x by μ.

[3] We consider dense real-valued time, so the set of possible durations for each uncontrollable action is infinite.

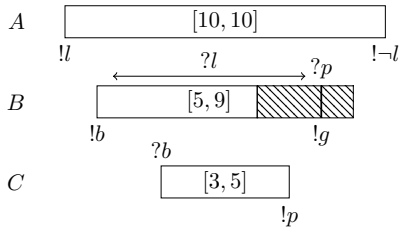

Figure 1: Example situation for which considering the upper or the lower bound on each uncontrollable action duration does not work. We indicate with $?l$, $!l$ the fact that literal l is a precondition, effect resp. for a given action.

the notion of validity for plans in temporal planning problems without uncertainty. Indeed, an STTP is valid if, for all possible durations of the uncontrollable actions, the plans resulting from considering such durations as fixed are valid, disregarding temporal uncertainty.

Definition 6. *An STTP σ is **valid** for (F, DA_c, DA_u, I, G) if and only if each $\pi \in \mathcal{I}_\sigma$ is a valid plan for the temporal planning problem $(F, DA_c \cup DA_u, I, G)$.*

It is easy to see that, if a strong planning problem with temporal uncertainty has no uncontrollable actions ($DA_u = \emptyset$), the semantics of plain temporal plans and STTP coincide.

The strong planning problem with temporal uncertainty is in some sense similar to a conformant planning problem (Ghallab, Nau, and Traverso 2004) when actions have non-deterministic effects. In fact, the duration of an action can be thought as part of the action effect. The difference is that here we consider dense real-valued time, so the set of possible durations is infinite, yielding an infinite set of possible non-deterministic effects. Nonetheless, we are aiming at a plan that is guaranteed to achieve the goal for every possible non-deterministic action duration.

At first sight, a possible and simple approach to solve a strong planning problem with temporal uncertainty could be to remove uncertainty in the domain, forcing either the upper or the lower bound on each uncontrollable action. Figure 1 shows an example in which this conjecture fails. In the example, action A sets l to true when it starts and to false when it terminates; action B is uncontrollable and requires l as an overall condition and requires p to be true to terminate, in addition it sets b to true when it starts and g to true when it ends. Finally, action C requires b to start and produces p as it terminates. The problem clearly admits a simple STTP: start A at time 0 with duration 10, B at time 0.5 and C at time 1 with duration 3. If we consider the plain temporal problem in which we fix the duration of B as its upper bound (9) we can get wrong assignments for C as it might be scheduled at time 3: in this case if B happens to last for only 5 time units, we violate the precondition of A, thus fixing the start of C at time 3 is not an STTP. Similarly, if we set the duration as the lower bound we can postpone the start of B in such a way that there exists an execution of B that violates its overall constraint. Nevertheless, there may be cases where this approach is sound, but this is out of the scope of this paper: the clear identification of these cases is left as future work.

Algorithm 2 FSSTP for strong planning problem with temporal uncertainty

```
1: procedure FSSTP(P)
2:     for all partial χ generated while solving abs(P) do
3:         X_u ← UncontrollableSteps(χ, P)
4:         D_c ← ControllableDurations(χ, P)
5:         D_u ← UncontrollableDurations(χ, P)
6:         P ← Precedences(χ, P)
7:         if μ ← TPU.Solve((χ − X_u, X_u, D_u, D_c ∪ P)) then
8:             if IsComplete(χ) then
9:                 return BuildSTTP(μ, χ)
10:            else continue( )
11:        else reject(χ)
12:    return ⊥
```

FSSTP with Temporal Uncertainty

In order to solve strong planning problems with temporal uncertainty, we propose a way to extend FSSTP. We retain the overall framework but we modify the scheduling check. The new framework is shown in Algorithm 2.

We first substitute the TP consistency check with the SC check of a TPU. As before, we use the steps of χ as time points for the temporal problem. The TPU is built by first separating the controllable steps in χ from the uncontrollable ones: for each uncontrollable action A, $A_\dashv$ is considered uncontrollable, while all the other steps are controllable. The duration constraints are built analogously to the plain temporal case, but are divided in two sets: D_c are the duration constraints for controllable actions, D_u are the ones for uncontrollable actions.

Building an STTP σ from a strong schedule for the TPU is analogous to the plain temporal planning case: each pair of snap actions $(A_\vdash, A_\dashv) \in \chi$ is a step of σ, the time for the step is $\mu(A_\vdash)$ and, if A is controllable, the duration is $\mu(A_\dashv) - \mu(A_\vdash)$. We do not set the duration for uncontrollable durative actions.

The encoding of the precedence constraints P is crucial, because in presence of uncontrollability not all the techniques presented in the temporal planning literature for the controllable case (Coles et al. 2012a; 2010) are complete. In the following, we consider two different encodings proposed in the temporal planning literature and we show that they are incomplete for solving the strong planning problem with temporal uncertainty. Then, we borrow the idea of reordering from Bäckström (1998) and we derive the first sound and complete approach for the strong planning problem with temporal uncertainty.

Total Order Encoding. A simple way of building P is to maintain the total order of the partial plan χ. Forcing this total order, clearly maintains the causal soundness but, as noted in (Coles et al. 2010), is heavily dependent on the order of actions chosen by the classical planner. Nevertheless, this encoding is complete for plain temporal planning and is adopted in the COLIN and CRIKEY 3 planners (Coles et al. 2012a). We call TO the set of precedence constraints for a given totally ordered plan $\chi = (a_1, \ldots, a_n)$, and we define it as follows:

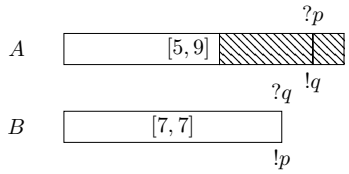

Figure 2: Example problem for which the TO and LAD encodings cannot find a plan, while an STTP exists.

$$TO \doteq \{a_i < a_{i+1} | 1 \leq i \leq n-1\}.$$

We highlight that no disjunction is created, hence the encoding results in an STPU. Despite its simplicity, this encoding is incomplete in presence of temporal uncertainty. Indeed, at each step of Algorithm 2, it might be the case that no total order produces a strongly controllable TPU, even if there exists an STTP for the given problem. Nonetheless, the approach is sound: if a solution is returned, it is a valid STTP.

As an example, let us consider the situation depicted in Figure 2. Let us suppose that both actions A and B must be started at the same time[4]. Action A is uncontrollable and B must end between the earliest and the latest possible ends of A. Literals p and q are initially true and no action falsify them. Let us focus on the relative order of $A_\dashv$ and $B_\dashv$. If $\chi = (\cdots, A_\dashv, \cdots, B_\dashv, \cdots)$, then we (transitively) impose the constraint $A_\dashv < B_\dashv$, but this makes the STPU not SC, because, if A takes longer than 7 time units, $A_\dashv$ can happen after $B_\dashv$ violating the constraint. If $\chi = (\cdots, B_\dashv, \cdots, A_\dashv, \cdots)$, then the situation is reversed and again the STPU is not SC. Therefore, in both cases χ is rejected and the planner returns $\perp$. This is incomplete, because there exists a simple STTP for the problem: start both actions at time 0 (the two actions are non-interfering and all the conditions are satisfied as p and q are never falsified).

Last Achiever Deordering Encoding. Another encoding that has been proposed, is to lift totally ordered plans to partially ordered plans (Coles et al. 2010). The underlying idea is to use the greedy algorithm by Veloso, Perez, and Carbonell (1990) to reconstruct the causal links as precedence links. For each action in the plan requiring a literal l as precondition, the algorithm searches for the last achiever of that literal in the totally ordered plan, and imposes a precedence link between the two actions. In this way, it builds a partial order plan as a deordering (Bäckström 1998) of χ and possibly reduces the commit on the specific input ordering. Also this encoding never introduces disjunctions, hence the resulting TPU is simple. Due to space constraints we do not report the full algorithm here, see Coles et al. (2010).

This encoding is able to find a plan in many situations even in presence of uncertainty, but it does not work in general. For example, it fails on the problem of Figure 2: the encoding greedily assumes that the last achiever is the one that must be preserved in the form of a causal link; in reality there may be other achievers that could be used instead. Just as in the previous case, if $\chi = (\cdots, A_\dashv, \cdots, B_\dashv, \cdots)$, then we impose the constraint $A_\dashv < B_\dashv$ because $A_\dashv$ is the last achiever of p (required by $B_\dashv$). Instead, if $\chi = (\cdots, B_\dashv, \cdots, A_\dashv, \cdots)$, we impose the constraint $B_\dashv < A_\dashv$ because $B_\dashv$ is the last achiever of q (required by $A_\dashv$).

Disjunctive Reordering Encoding. In order to obtain a sound and complete reasoning, we need to relax the total order produced by the state-space search, retaining the precedence constraints needed to ensure plan validity. However, we must be careful in not over-constraining the TPU, otherwise we may discard valid plans. A solution is to consider all the reorderings (Bäckström 1998) of the given plan that are causally sound: we build a set of (disjunctive) precedence constraints in such a way that all the orderings fulfilling the constraints are causally sound. We call *Disjunctive Reordering* (written DR) such a set of constraints. We show that, given a partial plan χ, using DR to construct the precedences in Algorithm 2, yields a complete technique for the strong planning problem with temporal uncertainty.

Given a literal l, we denote with $f(l)$ the set $\{a \in \chi | l \in eff(a)\}$ of actions that achieve l. Given a literal l and a pair of actions a and r, we define the temporal constraint $\rho(l, a, r) \doteq (a < r \wedge \bigwedge_{a_i \in f(l) \setminus \{a,r\}} (a_i < a \vee a_i > r))$. Intuitively, for a literal l, if a is an achiever of l and r is an action having l as precondition, $\rho(l, a, r)$ holds if a was the last achiever of l before r. We now define DR. Using a common trick in partial order planning, we consider two fictitious actions, a_0 and a_{n+1}, representing the initial state and the goal condition, respectively. Action a_0 has no preconditions and has the initial state I as effect. Action a_{n+1} has the goal as precondition and no effect.

Definition 7. *Given $\chi = (a_0, \cdots, a_{n+1})$, DR is as follows. For each $1 \leq i \leq n$, $\{(a_0 < a_i), (a_i < a_{n+1})\} \subseteq DR$. For each $a \in \chi$ and for each $l \in pre(a)$, the following constraints belong to DR:*

1. $\bigvee_{a_j \in f(l) \setminus \{a\}} \rho(l, a_j, a)$;
2. $\bigwedge_{a_j \in f(l)} (\rho(l, a_j, a) \rightarrow \bigwedge_{a_t \in f(\neg l) \setminus \{a\}} (a_t < a_j \vee a_t > a))$.

For each $a_\vdash \in \chi$ and for each $l \in overall(a)$, the following constraint is in DR:

3. $\bigwedge_{a_j \in f(\neg l)} ((a_j < a_\vdash) \vee (a_j > a_\dashv))$.

Intuitively, constraint 1 says that at least one action a_j having as effect the precondition l of action a, should occur before a. Constraint 2 says that, if a_j is the last achiever for the precondition l of action a, between a_j and a there must be no action falsifying l. Invariants are treated as preconditions that cannot be canceled until the end of the action (constraint 3). The following theorem states that DR is sound and complete. [5]

Theorem 1. *Given a strong planning problem with temporal uncertainty admitting a valid STTP σ, if DR is used, Algorithm 2 terminates with a valid STTP.*

The intuition is that in DR the disjunctions encode all reorderings that are causally sound in the form of a DTPU, allowing the scheduler to re-arrange the actions independently of the total ordering of χ.

[4] We just need that the end of B is forced to overlap with the interval in which A can uncontrollably end.

[5] The proof can be found in the additional material available at: https://es-static.fbk.eu/people/roveri/tests/aaai15.

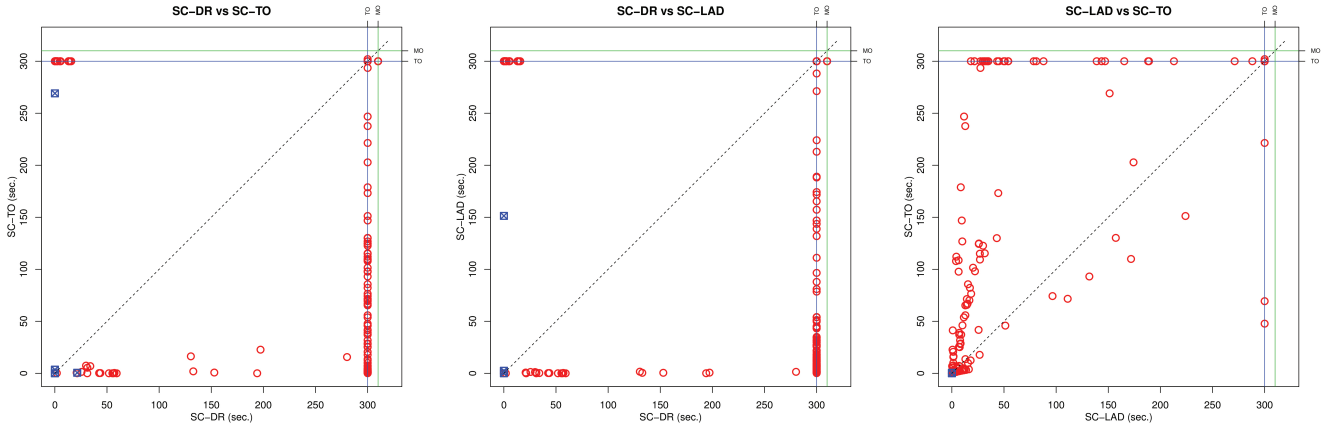

Figure 3: Results for the experimental evaluation. The "TO" and "MO"lines indicate the time-out and the memory-out respectively. Red circles are points where compared solvers agree, while blue squares are when they disagree, due to incompleteness.

Solving the SC of a DTPU is a NP-Hard problem (Peintner, Venable, and Yorke-Smith 2007), thus the use of this encoding is quite costly; however, DR is important as it overcomes the incompleteness limitation of the other encodings.

Implementation and Experiments

Implementation. We implemented the approaches described in the previous section as an extension of COLIN (Coles et al. 2012a). COLIN can handle temporal domains expressed in PDDL 2.1, and its temporal reasoning component is modular w.r.t. the rest of the code base: it has a clear notion of scheduler for checking temporal consistency.

The parser and the internal structures of COLIN were modified to support the processing of an extension of PDDL 2.1 with actions with uncontrollable durations. We added three new schedulers, each implementing one of the defined encodings. We write SC-TO to refer to the encoding based on total ordering, SC-LAD for the Last Achiever Deordering and SC-DR for the Disjunctive Reordering. The heuristic for the forward search planner was left unchanged. The temporal solvers for SC were implemented in C++, following the approach in (Cimatti, Micheli, and Roveri 2014), and using the MathSAT (Cimatti et al. 2013) SMT solver.

Setup. We considered all planning problems from the temporal track of the IPC 2011 (Coles et al. 2012b): we modified them by declaring some actions uncontrollable and by enlarging the duration intervals of actions. [6] We also analyzed the problems from Cimatti, Micheli, and Roveri (2013), and the examples presented in this paper. Overall, this resulted in a benchmark set of 563 problems. Given that there are no other planners able to deal with actions with uncontrollable durations, we compared the search time of our three approaches: SC-DR, SC-LAD, and SC-TO. The experiments were executed on a Scientific Linux 64 bit, 12 cores Intel Xeon at 2.67GHz, with 96GB RAM. We used a timeout of 5 minutes, and a memory limit of 8GB.

[6]The tool and the benchmark set can be downloaded from: https://es-static.fbk.eu/people/roveri/tests/aaai15.

Results. The results for the comparisons are reported in Figure 3. The left plot compares SC-DR and SC-TO; the center plot SC-DR and SC-LAD and the right plot SC-LAD and SC-TO. We recall that SC-DR is complete, while the others are not. Nonetheless, all the approaches are sound: if a plan is returned, it is valid. We observe that in most cases, the incomplete approaches are faster than SC-DR: the reason is that checking strong controllability of STPUs is much cheaper than checking strong controllability of DTPUs. However, in a number of points, SC-DR terminates where SC-LAD and SC-TO do not. These are the examples where they cannot build a plan due to incompleteness, while a valid STTP exists. The right plot compares the two incomplete techniques and shows that, although in few cases SC-TO is better than SC-LAD, there are many cases where SC-TO goes timeout while SC-LAD is able to find a solution in the time limits. This is because of the strong commitment of SC-TO to the specific order produced by the classical planner. In presence of uncontrollability it is often the case that SC-TO builds over-constrained TPUs that are rejected, while SC-LAD is able to accept the same partial plan thanks to its partial order. In fact, there are also some cases where SC-TO says that there is no solution, while SC-LAD is able to find an STTP. Hence, there is no pay off in using SC-TO instead of SC-LAD for problems with uncontrollable action durations. As side remark, we report that we solved the benchmarks of Cimatti, Micheli, and Roveri (2013) in less than a second instead of about two minutes.

Conclusions and future work

In this paper, we defined the problem of strong planning with temporal uncertainty, and we proposed the first sound and complete planning algorithm. We implemented the proposed approach, and we carried out an experimental evaluation showing its feasibility. As future work, we will investigate ways to mitigate the cost of the DR approach retaining its completeness, possibly by defining dedicated heuristics for the planner. In addition, we would like to extend our approach to handle numeric fluents and continuous change.

Acknowledgments

We would like to thank Michele Cristelli for the discussions on a preliminary version of this work and for the implementation of the TO approach in an old version of COLIN.

References

Bäckström, C. 1998. Computational aspects of reordering plans. *J. Artif. Intell. Res. (JAIR)* 9:99–137.

Barrett, C. W.; Sebastiani, R.; Seshia, S. A.; and Tinelli, C. 2009. Satisfiability modulo theories. In *Handbook of Satisfiability*. IOS Press. 825–885.

Cesta, A.; Cortellessa, G.; Fratini, S.; Oddi, A.; and Rasconi, R. 2009. The APSI Framework: a Planning and Scheduling Software Development Environment. In *Working Notes of the ICAPS-09 Application Showcase Program*.

Cimatti, A.; Griggio, A.; Schaafsma, B. J.; and Sebastiani, R. 2013. The MathSAT5 SMT Solver. In Piterman, N., and Smolka, S. A., eds., *TACAS*, volume 7795 of *LNCS*, 93–107. Springer.

Cimatti, A.; Micheli, A.; and Roveri, M. 2013. Timelines with temporal uncertainty. In desJardins, M., and Littman, M. L., eds., *AAAI*, 195–201. AAAI Press.

Cimatti, A.; Micheli, A.; and Roveri, M. 2014. Solving strong controllability of temporal problems with uncertainty using SMT. *Constraints* 1–29.

Coles, A.; Fox, M.; Long, D.; and Smith, A. 2008. Planning with problems requiring temporal coordination. In *AAAI*, 892–897.

Coles, A.; Fox, M.; Halsey, K.; Long, D.; and Smith, A. 2009. Managing concurrency in temporal planning using planner-scheduler interaction. *Artif. Intell.* 173(1):1–44.

Coles, A. J.; Coles, A.; Fox, M.; and Long, D. 2010. Forward-chaining partial-order planning. In *ICAPS*, 42–49.

Coles, A. J.; Coles, A.; Fox, M.; and Long, D. 2012a. Colin: Planning with continuous linear numeric change. *J. Artif. Intell. Res. (JAIR)* 44:1–96.

Coles, A. J.; Coles, A.; Olaya, A. G.; Celorrio, S. J.; López, C. L.; Sanner, S.; and Yoon, S. 2012b. A survey of the seventh international planning competition. *AI Magazine* 33(1).

Cushing, W.; Kambhampati, S.; Mausam; and Weld, D. S. 2007. When is temporal planning really temporal? In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, 1852–1859.

Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal constraint networks. *Artif. Intell.* 49(1-3):61–95.

Fox, M., and Long, D. 2003. PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains. *J. Artif. Intell. Res. (JAIR)* 20:61–124.

Frank, J., and Jónsson, A. 2003. Constraint-based Attribute and Interval Planning. *Constraints* 8(4):339–364.

Ghallab, M.; Nau, D. S.; and Traverso, P. 2004. *Automated planning - theory and practice*. Elsevier.

Muise, C. J.; Beck, J. C.; and McIlraith, S. A. 2013. Flexible execution of partial order plans with temporal constraints. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*.

Peintner, B.; Venable, K. B.; and Yorke-Smith, N. 2007. Strong controllability of disjunctive temporal problems with uncertainty. In *CP*, 856–863.

Penberthy, J. S., and Weld, D. S. 1994. Temporal planning with continuous change. In *AAAI*, 1010–1015.

Tsamardinos, I., and Pollack, M. E. 2003. Efficient solution techniques for disjunctive temporal reasoning problems. *Artificial Intelligence* 151(12):43 – 89.

Veloso, M. M.; Perez, M. A.; and Carbonell, J. G. 1990. Nonlinear planning with parallel resource allocation. In *Proceedings of the DARPA Workshop on Innovative Approaches to Planning, Scheduling, and Control*, 207–212. Morgan Kaufmann.

Vidal, T., and Fargier, H. 1999. Handling contingency in temporal constraint networks: from consistency to controllabilities. *J. Exp. Theor. Artif. Intell.* 11(1):23–45.

Wilson, M.; Klos, T.; Witteveen, C.; and Huisman, B. 2014. Flexibility and decoupling in simple temporal networks. *Artificial Intelligence* 214(0):26 – 44.

Younes, H. L. S., and Simmons, R. G. 2003. VHPOP: versatile heuristic partial order planner. *J. Artif. Intell. Res. (JAIR)* 20:405–430.