Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling

Temporal Landmarks: What Must Happen, and When

Erez Karpas, David Wang, Brian C. Williams
Massachusetts Institute of Technology

Patrik Haslum
Australian National University & NICTA

Abstract

Current temporal planners have a hard time solving large, real-world problems which involve dealing with metric time and concurrent actions. While landmarks have enabled classical planners to scale up to significantly larger problems, they have not yet brought as much benefit to temporal planning. We argue that the reason for this is that for landmarks to make an effective addition to planning with complex temporal interactions (such as required concurrency), they must incorporate information about the timing of conditions and events. We define temporal landmarks, which associate time intervals and time points, respectively, with state and action landmarks, thereby capturing both what must happen and when it must happen. We show how to derive temporal landmarks and constraints on their associated time points from planning problems, and how exploiting them, in a planner-independent way, can improve planner performance. Notably, the greatest gain is on problems which require concurrency, showing that the temporal information we add to landmarks complements the reasoning used by current temporal planners.

Introduction

Many real-world problems require handling complex temporal interactions which involve concurrently executing actions. However, current temporal planners have a hard time scaling up to large problems which require concurrency (Cushing, Kambhampati, and Weld 2007). One method which has enabled classical planners to scale up to significantly larger problems is the use of landmarks. In classical planning, a landmark is a fact which must be achieved at some point along every solution (Hoffmann, Porteous, and Sebastia 2004).

In temporal planning, one line of work has also used the notion of landmarks (Sebastia, Marzal, and Onaindia 2007; Marzal, Sebastia, and Onaindia 2008; 2014). This work starts by discovering the causal (non-temporal) landmarks of the problem, and then exploits deadlines to tighten temporal constraints on when each landmark must be achieved, and infer further landmarks. However, when there are no deadlines in the problem, this approach does not yield any benefit over the causal landmarks.

In this paper, we define a novel notion of *temporal landmarks*, which describe both *what* must happen and *when* this must happen. Specifically, we define two types of landmarks: temporal fact landmarks, which state that some fact must hold between two timepoints, and temporal action landmarks, which state that some event (the start or end of an action) must occur at some time point. Temporal information is captured in the form of simple temporal constraints over the *symbolic* time points (Dechter, Meiri, and Pearl 1991) associated with each landmark. Thus a set of landmarks and constraints constitutes a qualitative state plan, or QSP (Hofmann and Williams 2006).

The contributions of this paper are threefold: First, we introduce a novel notion of temporal landmarks. Second, we describe a sound technique for discovering temporal landmarks using a backchaining approach and a set of derivation rules. Third, we demonstrate the usefulness of temporal landmarks, both theoretically on an example problem, and empirically on a set of IPC benchmarks. Our empirical evaluation shows that temporal landmarks allow temporal planners to scale up to larger problems which require concurrency, and can also help in problems without required concurrency.

Notation

We use the non-numeric subset of PDDL 2.1 for expressing temporal planning problems (Fox and Long 2003). In this section we review the relevant definitions and notations for describing such problems. A *propositional temporal planning problem* is a tuple $\Pi = \langle F, A, I, G \rangle$ where F is a set of propositions, $I \subseteq F$ is the initial state, $G \subseteq F$ is the goal, and A is a set of *actions*. An *event* e is described by a tuple $\langle \mathrm{cnd}, \mathrm{eff} \rangle$, where $\mathrm{cnd} \subseteq F$ is a precondition, which must hold in the state right before the event is executed, and eff is an effect on F, which occurs right after the event is executed. An instantaneous action consists of a single event $\langle \mathrm{cnd}(a), \mathrm{eff}(a) \rangle$. A durative action consists of a start event, $start(a) = \langle \mathrm{cnd}_s(a), \mathrm{eff}_s(a) \rangle$, an end event $end(a) = \langle \mathrm{cnd}_e(a), \mathrm{eff}_e(a) \rangle$, an invariant condition $\mathrm{cnd}_i(a)$ which must hold throughout the execution of the action, and a minimum and maximum duration, $d_{\min}(a)$ and $d_{\max}(a)$.

A schedule for Π is a list of triples, where each such triple $\langle t, d, a \rangle$ denotes that action a starts at time t and lasts for a duration of d. A solution for Π is a schedule such that the goal G holds after all actions have completed execution, starting from the initial state I. We furthermore require that pairs of events which are mutually exclusive, or where one achieves some proposition that the other requires, are scheduled at least some $\epsilon > 0$ time units apart.

Given an event e which is the start/end of a durative action a, we refer to the event at the opposite end (that is, the end/start of the same action) by $oe(e)$. If e is the single event of an instantaneous action a, then $oe(e) = e$.

We will also use a few symbolic time points:

- t_{START} is the time when execution starts, and we assume that $t_{START} = 0$
- t_{END} is the time when the last action ends, and thus the state stops changing after t_{END}

Temporal Landmarks

Although in classical planning, fact landmarks and action landmarks are often used interchangeably, this is not so easy in temporal planning. The main reason for this difference is that in temporal planning, states hold for some length of time, while actions start and end instantaneously. Therefore, temporal fact landmarks and temporal action landmarks differ more than they do in classical planning.

Instead of action landmarks consisting of sets of actions, a temporal action landmark consists of a set of events (start/end of actions). Additionally, we associate with each temporal action landmark the time point in which one of these events must occur. We denote such a landmark by $occurs_t(E)$, meaning that one of the events $e \in E$ must occur at time point t.

Definition 1 *A* temporal action landmark $occurs_t(E)$ *consists of a set of events E, and a time point t, denoting that one of the events in E must occur at time point t.*

Note that since a temporal action landmark refers to a single time point, using a conjunction, meaning that two events occur at this time point, would require perfect synchronization. As the execution semantics discourage such solutions, using conjunctive temporal action landmarks would require us to associate a different time point with each event, which would complicate the management of temporal constraints.

On the other hand, states hold over a duration, and therefore, for temporal fact landmarks, we consider an arbitrary formula over the propositions of the planning task, which must hold for some duration. Informally, $holds_{t_s:t_e}(\Phi)$ means that formula Φ must hold for a duration of time, starting *exactly* at time point t_s, and holding *at least* until time point t_e.

Definition 2 *A* temporal fact landmark $holds_{t_s:t_e}(\Phi)$ *consists of a boolean formula Φ over the propositions of* Π*, and two time points t_s and t_e, denoting the time when Φ becomes true, and a time when Φ is no longer required to hold, respectively.*

We also maintain a set of simple temporal constraints between the time points associated with the landmarks, which thus constitute a Simple Temporal Network (Dechter, Meiri, and Pearl 1991). Simple temporal constraints are of the form $l \leq t_2 - t_1 \leq u$, where t_1 and t_2 are time points, and $l, u \in \mathbb{R} \cup \{-\infty, \infty\}$ are used to bound the separation between them. Note that by using t_{START} in a simple temporal constraint, we can specify an absolute range for when a time point occurs.

Using this more expressive language to describe landmarks allows us to express constraints about solutions which are impossible to express using "regular" landmarks. For example, it is possible to specify that some proposition p must be achieved twice, by stating that $holds_{t_1:t_2}(p)$ and $holds_{t_3:t_4}(p)$ are landmarks, with $t_2 < t_3$. Note that had we defined the starting point of a temporal fact landmark to denote a time when the formula holds, rather than a time when the formula *becomes* true, this would not have been the case. Additionally, this more expressive language allows us to revisit the notion of orderings between landmarks, and instead of just specifying that some landmark must occur before another, we can also specify some range on the duration between these events.

We conclude this section by giving a formal definition for when a set of landmarks and constraints is *valid* with regards to a planning task — that is, the landmarks and constraints hold in all possible solutions.

Definition 3 *A set of temporal landmarks $\mathcal{L}$ and constraints $\mathcal{C}$ involving time points TV is* valid *with regards to planning task* Π*, iff for every schedule τ for* Π*, there exists a mapping $m : TV \to \mathbb{R}^{0+}$, such that:*

- *Mapping m respects the constraints $\mathcal{C}$, and*
- *For any temporal action landmark $occurs_t(E) \in \mathcal{L}$, there exists some event $e \in E$ such that τ schedules e at time $m(t)$, and*
- *For any temporal fact landmark $holds_{t_s:t_e}(\Phi)$, formula Φ holds during the execution of τ in the interval $[m(t_s), m(t_e)]$, and if $m(t_s) > 0$ then Φ becomes true at time $m(t_s)$, i.e. there exists some $\delta > 0$ such that for all $\epsilon \in (0, \delta)$, Φ does not hold at time $m(t_s) - \epsilon$.*

The set of fact landmarks and constraints can be viewed as a qualitative state plan (QSP) (Hofmann and Williams 2006), where the fact landmarks and temporal constraints specify the desired evolution of state over time. Action landmarks provide extra information about how this desired state trajectory can be achieved.

Temporal Landmarks Example

In the previous section, we described temporal landmarks, which allow us to assert claims about both *what* must happen and *when* it must happen. In the next section, we will describe a set of rules for deriving temporal landmarks. However, we will first demonstrate how temporal landmarks can be used to reason over a modified matchcellar problem, so that the landmarks completely characterize the solution. Throughout this example we also include pointers to the derivation rules in the next section, which formally characterize the reasoning process we use here.

- Propositions: $\{hM, hF, light, fixed\}$
- Initial state: $\{hM\}$
- Goal: $\{fixed\}$
- Actions:

Name	Duration	cnd$_s$	eff$_s$	cnd$_i$	cnd$_e$	eff$_e$
fix-fuse	10	$light$		$light$		fixed
light-match	5	hM	$light, \neg hM$			$\neg light$
find-flashlight	2	$light$		$light$		hF
turn-on-flashlight	1	hF				$light$

Figure 1: Description of Modified Matchcellar Task

The goal of our planning problem is to fix a fuse, an action which takes 10 time units and requires light throughout. We can light a match, which provides light for only 5 time units. However, there is also a flashlight somewhere, which can provide light indefinitely, after it is found. The flashlight can only be found if there is light, which takes 2 time units. This problem is described precisely in Figure 1.

If we do not take into account the durations of actions, and discover only the causal landmarks of this problem, we would get $fixed$, $light$, and hM. This does not allow us to deduce that we must find the flashlight, as the match will not give light for enough time.

However, using our temporal landmarks, we could start with a landmark describing that the goal holds from some time point t_g, until the end of excution, that is $holds_{t_g:t_{END}}(fixed)$. The only achiever of $fixed$ is end(fix-fuse), and thus $occurs_{t_{eff}}(end(\text{fix-fuse}))$ is also a temporal landmark, with constraint $t_{eff} = t_g$ (this is derivation rule I in the next section).

Every action that ends must start, and therefore $occurs_{t_{sff}}(start(\text{fix-fuse}))$ is also a temporal landmark, with the constraint $t_{eff} = t_{sff} + 10$. The invariant condition of the action must also hold throughout the execution, and therefore $holds_{t_{sl}:t_{el}}(light)$ is a temporal landmark, with the constraints $t_{sl} \leq t_{sff}$ and $t_{el} = t_{eff} - \epsilon$ (these are both described in rule V in the next section).

There are two possible achievers for $light$: end(turn-on-flashlight), and $start$(light-match). However, we can easily show that $t_{el} - t_{sl} \geq 10 - \epsilon$. As the duration of light-match is 5, and it deletes $light$ at the end, light-match can not be used to achieve $light$ for $10 - \epsilon$ time units, and thus we can eliminate it as an achiever and deduce that $occurs_{t_{etof}}(end(\text{turn-on-flashlight}))$ is a temporal landmark, with $t_{etof} = t_{sl}$ (rule I). This also entails that $occurs_{t_{stof}}(start(\text{turn-on-flashlight}))$ is a temporal landmark, with $t_{stof} = t_{etof} - 1$ (rule V). The start condition of turn-on-flashlight gives us $holds_{t_{shf}:t_{ehf}}(hF)$, with $t_{shf} < t_{stof}$ and $t_{ehf} \geq t_{stof}$ (rule III).

The only possible achiever of hF is find-flashlight, and thus $occurs_{t_{ef}}(end(\text{find-flashlight}))$, with $t_{ef} = t_{shf}$ (rule I), and $occurs_{t_{sf}}(start(\text{find-flashlight}))$, with $t_{sf} = t_{ef} - 2$ (rule V) are temporal landmarks. This also gives us the invariant condition landmark $holds_{t_{slf}:t_{elf}}(light)$, with $t_{slf} \leq t_{sf}$ and $t_{elf} = t_{ef} - \epsilon$ (also rule V).

From $holds_{t_{slf}:t_{elf}}(light)$ we can then derive the temporal landmark consisting of the first time possible achievers of $light$. As the flashlight can not provide light before we find it, which requires light, $occurs_{t_{slm}}(start(\text{light-match}))$ is also a temporal landmark, with $t_{slm} \leq t_{slf}$ (rule II).

Thus, we have accounted for all of the actions in the solution, using only reasoning about temporal landmarks. Figure 2 shows these time points and the constraints connecting them graphically.

Temporal Landmark Extraction

Having defined a language for describing temporal landmarks and demonstrating that it can be useful on an example problem, we must still provide a general method for discovering temporal landmarks in any given planning problem. This is a computationally hard problem even in classical planning (Hoffmann, Porteous, and Sebastia 2004), and therefore we focus on a tractable algorithm which might not discover all temporal landmarks. Our landmark extraction technique is similar to backchaining approaches used in landmark extraction for classical planning (Hoffmann, Porteous, and Sebastia 2004; Richter and Westphal 2010), but also reasons over the temporal information associated with each landmark.

Initial Landmark

In order to use backchaining, we must start with some set of trivial landmarks. One such trivial landmark is the goal, which must obviously hold in every solution. We will define the time point t_G as the *last* time in which the goal was achieved (that is, the value of the goal formula transitions from false to true). By definition of t_G, once the goal is achieved at t_G it stays true forever.

Using t_G and the previously defined t_{END} (the last time point in which any event can occur), we can formulate the landmark $holds_{t_G:t_{END}}(G)$ — the goal must hold from t_G until the end of execution. We can also derive the temporal constraint $t_G \leq t_{END}$. Furthermore, if the goal has a deadline t, we can add the constraint $t_G - t_{START} \leq t$.

Another option is to start backchaining from a set of non-temporal landmarks, as done by Marzal, Sebastia, and Onaindia. We can take the classical relaxation of our temporal problem (Haslum 2009), and apply a classical landmark detection technique to it. Any landmark Φ of the classical relaxation yields a temporal fact landmark $holds_{t_s:t_e}(\Phi)$, with the constraints $t_s \geq t_{START}$ and $t_e \leq t_{END}$.

With a set of initial landmarks, we can start backchaining using a set of derivation rules. The following two subsections describe a set of such rules, which allow us to discover more landmarks.

Backchaining from Temporal Fact Landmarks

If $L = holds_{t_s:t_e}(\Phi)$ is a temporal fact landmark, and Φ is not satisfied by the initial state, then we can derive temporal landmarks according to the following rules:

I $occurs_{t_a}(E)$ is a temporal landmark, where $E = pa_{UB(t_s)}(\Phi) \setminus ineligible(d, \Phi)$, $pa_t(\Phi)$ is a superset of the events which possibly achieve Φ by time t (we describe how to obtain such a set later in this paper), from which we eliminate some ineligible achievers, $ineligible(d, \Phi) := \{e \mid e = start(a), d_{\max}(a) < d,$ and $\text{eff}_e(a) \models \neg\Phi\}$ — start events that could make

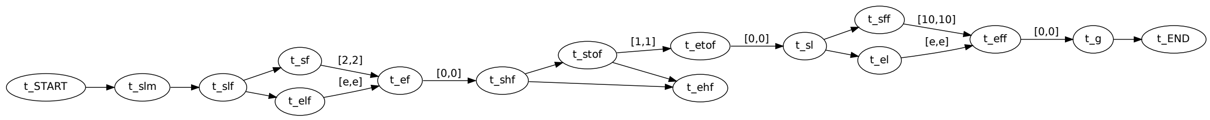

Figure 2: Time Points and Constraints of Temporal Landmarks

Φ true, but belong to an action whose end effect will then delete Φ (that is, there is no way for Φ to hold after the end effect of a has executed), with durations that are shorter than the minimal duration for which Φ must hold. $UB(t)$ is an upper-bound on t and $d = LB(t_e - t_s)$ is a lower-bound on $t_e - t_s$. Both of these bounds are the tightest implied by the current temporal constraints. As these constraints are all simple temporal constraints, we can compute these bounds efficiently (Dechter, Meiri, and Pearl 1991).

Additionally, we add the constraint $t_a = t_s$. This is the main reason for defining t_s as the time when Φ *becomes* true, rather than as some time from which we know Φ to be true. Using the weaker definition, where t_s is some time where Φ holds, we would only be able to derive $t_a < t_s$. If the only possible achievers of Φ are start events of actions which also delete Φ at the end, we can derive even more constraints: we require that the action that achieves Φ end after t_e, and thus can derive the constraint $t_a + \max_{start(a)\in E} d_{\max}(a) \geq t_e$.

II $occurs_{t'_a}(fa_{UB(t_s)}(\Phi))$ is a temporal landmark, where $fa_t(\Phi)$ is a superset of the events which possibly achieve Φ *for the first time* by time t. We can also infer that $t'_a \leq t_s$. Note that, unlike the previous case, we can not subtract $ineligible(d, \Phi)$ from the first achievers, as we do not know whether L refers to the first time Φ is achieved or not, and it is possible that the first time Φ was achieved was for a shorter duration than d. However, we do know that if Φ is achieved, then it must be achieved for the first time. Note that as we restrict our attention to only *first* achievers — a more restrictive set of actions — we might get a stronger landmark than the previous rule yields, even though the temporal constraints are weaker.

Backchaining from Temporal Action Landmarks

From a temporal action landmark $L = occurs_{t_a}(E)$, we can also derive more temporal landmarks, according to the following rules:

III $holds_{t_s:t_e}(\text{cc}(\{\text{cnd}(e) \mid e \in E\}))$, where

$$\text{cnd}(e) = \begin{cases} \text{cnd}_s(a) & e = start(a) \\ \text{cnd}_e(a) & e = end(a) \end{cases}$$

$\text{cc}(\{\Phi_1 \dots \Phi_n\})$ is a common (logical) consequence of *all* of $\Phi_1 \dots \Phi_n$, that is, a fact implied by Φ_1, and by Φ_2, ... and by Φ_n. In other words, $\text{cc}(\{\text{cnd}(e) \mid e \in E\})$ is a common condition for all events in E. In case $\Phi_1 \dots \Phi_n$ are conjunctions over facts, the strongest common consequent is the conjunction of all literals that appear in each of $\Phi_1 \dots \Phi_n$. Furthermore, this condition must hold when the event occurs, giving us the constraints $t_s < t_a$ and $t_e \geq t_a$. The first constraint is a strict inequality due to the "no moving targets" rule (Fox and Long 2003).

IV $holds_{t_s:t_e}(\text{cc}(\{\text{eff}(e) \mid e \in E\}))$, where

$$\text{eff}(e) = \begin{cases} \text{eff}_s(a) & e = start(a) \\ \text{eff}_e(a) & e = end(a) \end{cases}$$

That is, the common effect of all events in E must hold. If this common effect is mutex with the common condition on these events, then we also know that the common effect starts at time t_a, and get the constraint $t_s = t_a$. Otherwise, it might be the case that $\text{cc}(\{\text{eff}(e) \mid e \in E\})$ was already true, and thus we can not derive a temporal constraint relating t_a and t_s. Of course, in any case we have $t_s < t_e$.

V $occurs_{t'_a}(E')$ and $holds_{t_s:t_e}(\text{cc}(\{\text{cnd}_i(e) \mid e \in E\}))$, where $E' = \{oe(e) \mid e \in E\}$ and $\text{cnd}_i(e)$ is the invariant condition of the action associated with e. That is, if one of the events in E must occur, then so must the event at the other end of the action. Additionally, the common invariant condition of all actions in E must hold during the execution of the action. To derive constraints on t'_a, t_s, and t_e, we consider the following cases:

- If E consists only of start events of actions, then the end must occur after at least enough time for the shortest action in E to have completed has passed, but not after the maximum duration of the longest action could pass. That is, $\max_{start(a)\in E} d_{\max}(a) \geq t'_a - t_a \geq \min_{start(a)\in E} d_{\min}(a)$. The invariant condition must hold right after t_a, and therefore it must have started by t_a, and thus $t_s \leq t_a$. Finally, the invariant must hold until right before the action ends. As we can not express this exactly, we use an arbitrarily small ϵ, and add the slightly looser constraint $t_e = t'_a - \epsilon$.
- If E consists only of end events of actions, then we can derive $\max_{end(a)\in E} d_{\max}(a) \geq t_a - t'_a \geq \min_{end(a)\in E} d_{\min}(a)$, with $t_s \leq t'_a$ and $t_e = t_a - \epsilon$.
- Finally, if E consists of both start and end events of actions, then we can only use the bounds from the maximum durations in the above cases, and get the constraints $\max_{end(a)\in E} d_{\max}(a) \geq t_a - t'_a$ and

$\max_{start(a)\in E} d_{\max}(a) \geq t'_a - t_a$. For the invariant, we can only derive constraints from the minimum duration of actions in E: the invariant must last for a minimum duration, that is, $t_e - t_s \geq \min_{start(a)\in E \cup end(a)\in E} d_{\min}(a)$. We also know that the action starts at $\min(t_a, t'_a)$, so $t_s \leq t_a$ and $t_s \leq t'_a$.

Computing Possible Achievers

Recall that $pa_t(\Phi)$ (respectively, $fa_t(\Phi)$) is the set of events which possibly achieve Φ before time t (respectively, possibly achieve Φ *for the first time* before time t). Finding these sets exactly is computationally infeasible even for classical planning (Hoffmann, Porteous, and Sebastia 2004), and so we use an overapproximation of these.

Similarly to classical planning, we can find the possible achievers of Φ by looking at events (start/end of actions) with effects which might achieve Φ. In the simple case where Φ is a disjunction over facts, $pa_t(\Phi)$ is the set of events which have one of the disjuncts of Φ as an effect. The case of conjunctions can be handled similarly to how achievers of conjunctive landmarks are found (Keyder, Richter, and Helmert 2010).

If t is not unbounded (recall that t is the tightest upper bound we can derive on some time point), we further restrict this set of events to only those which could occur by time t according to the temporal relaxed planning graph (Coles et al. 2008). That is, we build the temporal relaxed planning graph until time t, and look at the possible achievers of Φ which are present there. Computing possible first time achievers, $fa_t(\Phi)$, can also be done similarly to classical planning. We construct the temporal relaxed planning graph up to time t, except that we do not use any action which adds Φ. The achievers of Φ which appear by time t are a superset of the possible first achievers of Φ.

Stopping Backchaining

So far, we have specified where backchaining starts, and a set of backchaining rules. We still need to specify when backchaining stops. With classical landmarks, backchaining terminates when it derives a landmark which has already been discovered or is true in the initial state.

With temporal landmarks, we do not necessarily have to stop backchaining when an already known landmark is discovered. This is because, as previously demonstrated, temporal landmarks are expressive enough to state that some landmark must be achieved twice, with different time points.

It is still possible to stop backchaining as soon as we derive a known landmark, where landmarks are compared without checking their time points. That is, if we already have $L = holds_{t_s:t_e}(\Phi)$ and we derive $L' = holds_{t'_s:t'_e}(\Phi)$, we would not add L', and stop backchaining. However, this could potentially lose some information from the temporal constraints between t'_s and t'_e and other time points.

Another option is to stop backchaining as soon as a loop is discovered in a single chain. During the backchaining process we keep track of the "parent" landmarks along the current chain. If the same landmark (again, ignoring time points) is discovered *along the same chain*, we stop backchaining. The potential drawback of this approach is that it might yield multiple landmarks, with different time points, which still describe the same thing. For example, the same occurrence of an action could satisfy multiple temporal action landmarks.

Refining Temporal Bounds

When deriving a temporal landmark, we can further refine the temporal constraints using information from the temporal relaxed planning graph. Given a temporal action landmark $occurs_t(E)$, we can add the constraint $t \geq \min_{e\in E} trpg(e)$, where $trpg(e)$ is the time in which event e appears in the TRPG for the first time. Similarly, for a temporal fact landmark $holds_{t_s:t_e}(\Phi)$ we can add the constraint $t_s \geq \min_i trpg(\Phi)$, where $trpg(\Phi)$ is the timestamp of the first layer where Φ holds.

Efficient Implementation

While the derivation rules above are described generally, in order to make reasoning about temporal landmarks more efficient, we restrict the formulas we use in temporal fact landmarks to be disjunctions over propositions. Thus we start backchaining from a set of goal landmarks, one for each proposition in the goal. It is important to note that this also means that instead of having a single time point, t_G, we have a separate time point t_g for each goal proposition $g \in G$.

Additionally, derivation rules which yield a temporal fact landmark with a conjunction instead produce several landmarks, one for each conjunct. Note that care must be taken when splitting a temporal fact landmark with a conjunction, $holds_{t_s:t_e}(\Phi_1 \wedge \ldots \wedge \Phi_n)$, into a set of landmarks, one for each conjunct. Specifically, it would not be correct to create the landmarks $holds_{t_s:t_e}(\Phi_1), \ldots, holds_{t_s:t_e}(\Phi_n)$, as only the conjunction $\Phi_1 \wedge \ldots \wedge \Phi_n$ is achieved *exactly* at t_s, not necessarily each of the conjuncts. Rather, each of the landmarks for the conjuncts would need to have its start time upper bounded by t_s. As the derivation rules described above already produce fact landmarks which only have an upper bound on their start times, we do not need to modify our derivation rules. However, should a new derivation rule, which produces a fact landmark with a lower bound on its start time be added, we would need to modify it accordingly.

Previous Temporal Landmark Techniques

Now that we have fully explained the formalism for our temporal landmarks, and how they are derived, we can compare our approach to previous work on temporal landmarks (Marzal, Sebastia, and Onaindia 2014). First, as the theoretical example illustrates, our approach does not rely on the presence of deadlines to discover landmarks that are not causal landmarks. On the other hand, Marzal, Sebastia, and Onaindia's approach only discovers non-causal landmarks when deadlines are present. Furthermore, even had there been a deadline on when the fuse must be fixed, Marzal, Sebastia, and Onaindia's approach would not be able to deduce that the flashlight must be found, as that is not a consequence of the goal deadline, but rather of *implicit* temporal constraints due to the duration a match can burn for.

On a more technical level, our approach is based on symbolic time points with simple temporal constraints between them. Marzal, Sebastia, and Onaindia's approach, on the other hand, associates with each landmark explicit intervals describing when it can become true, when it can be true, and when it must be true. Thus, while our formalism allows us to express the fact that some event must occur exactly 10 seconds before another event, even if there is no exact time for when either of these occurs, this is not possible in Marzal, Sebastia, and Onaindia's formalism.

Nevertheless, while both approaches differ on a technical level, they share many of the same intuitions. For example, taking fact landmark $holds_{t_s:t_e}(\Phi)$, t_s corresponds conceptually to the start of the validity interval of Φ, and t_e to the end of its necessity interval in Marzal, Sebastia, and Onaindia's work. Furthermore, Marzal, Sebastia, and Onaindia present some ideas which could be adapted to our framework, such as exploiting information about mutual exclution to further refine temporal constraints. Merging these two lines of research is a promising direction for future work.

Using Temporal Landmarks

Having discovered a set of temporal landmarks, we would like to use them in order to help guide a planner. We now briefly describe a few options for doing this.

Heuristics based on Temporal Landmarks

Temporal landmarks can provide us with information about what must occur in the future. As done with heuristics based on classical landmarks, we can look at the set of actions which must still occur in the future, given the landmarks that were already achieved, similarly to what LAMA does by looking at the possible achievers of each landmark (Richter and Westphal 2010). From this information, we can derive heuristic estimates over several different metrics.

First, we can simply count the number of actions that must still be applied, to get an estimate of the remaining plan length. To get an estimate on remaining cost, we can either sum over the costs of these actions, or derive a lower bound on remaining cost by using action cost partitioning (Karpas and Domshlak 2009). Another option is to use the heuristic described by Marzal, Sebastia, and Onaindia (2014).

Finally, if we want to derive an admissible estimate on makespan, we can create a simple temporal problem (Dechter, Meiri, and Pearl 1991), encoding the known temporal constraints. Solving this problem, which can be done efficiently, would give us a lower bound on t_{END}, which constitutes an admissible estimate of makespan.

Note that, as in LAMA, these are path-dependent heuristics. However, this is not a problem in the context of a heuristic forward search planner, as it has the path to each node it evaluates.

QSP and Constraint Based Planners

As previously mentioned, the set of temporal landmarks and constraints can be viewed as a qualitative state plan (Hofmann and Williams 2006). While this is not directly useful for most planners, the tBurton planner (Wang and Williams 2015) takes a QSP as its goal representation. Thus, we can use the temporal landmarks and constraints as tBurton's goal.

A potentially more powerful technique exploits the fact that tBurton searches over partial plans, which are all represented as QSPs. Thus, we can apply the temporal landmark backchaining rules whenever tBurton adds an action to the QSP. Furthermore, we can apply the backchaining rules only to the new action, rather than to the entire problem, thus perfoming more reasoning during tBurton's search. We intend to explore this technique in future work.

Additionally, partial order planners such as CPT (Vidal and Geffner 2006) can be initialized with a partial plan which corresponds to the temporal landmarks and constraints. CPT can then refine this partial plan into a full solution, without having to expend effot on deriving it initially.

Compiling Temporal Landmarks into the Problem

Another method of exploiting temporal landmarks, which can be used to endow any temporal planner with knowledge of the discovered temporal landmarks, is to compile them into the problem, similarly to previous work on compiling classical planning landmarks for use with abstraction heuristics (Domshlak, Katz, and Lefler 2012). Unfortunately, we do not have a good way of compiling the temporal constraints into the problem, only the temporal landmarks.

While the compilation is fairly straightforward, we prefer to modify the operator definitions at the PDDL domain level, rather than generate a fully grounded problem. This slightly complicates our compilation, which is described next:

- For every operator O, with parameters $o_1 \dots o_n$, such that a grounded action a derived from O appears in a landmark, we add two new predicates $started_O$ and $ended_O$, both with parameters $o_1 \dots o_n$. These are added at the start and end of O, respectively.
- For every predicate P, with parameters $o_1 \dots o_n$, such that a grounded proposition p derived from P appears in a landmark, we add a new predicate, $achieved_P$, also with parameters $o_1 \dots o_n$. Every operator that adds $P(o_1 \dots o_n)$ is modified to also add $achieved_P(o_1 \dots o_n)$, at the same time (that is, start or end) that it adds $P(o_1 \dots o_n)$.
- For every temporal landmark that consists of a single proposition or event, we add the corresponding new fact to the goal.
- For every disjunctive landmark L, we create a new proposition (that is, predicate with 0 arguments) $achieved_L()$, and add it to the goal. Additionally, for each proposition p that corresponds to each of the disjuncts in L, we add a new action that has precondition p and adds $achieved_L()$, with 0 cost and duration. Note that a temporal action landmark is a disjunction over its events, and so the propositions in its actions would be the appropriate $started_O$ or $ended_O$.

Note that compiliation of single (fact or action) landmarks can be done mostly on the symbolic level, by modifying

Domain	Solved				IPC Score			
	orig	e1	e4	e∞	orig	e1	e4	e∞
POPF2 (IPC-2011 Version)								
MATCHCELLAR (2011)	20	20	20	20	20.00	20.00	20.00	20.00
MATCHCELLAR (2014)	20	20	20	20	20.00	20.00	20.00	20.00
TMS (2011)	5	**11**	4	4	2.78	**11.00**	2.70	2.70
TMS (2014)	0	**6**	0	0	0.00	**6.00**	0.00	0.00
TURNANDOPEN (2011)	**9**	8	**9**	8	**8.47**	7.58	8.40	7.42
TURNANDOPEN (2014)	0	0	0	0	0.00	0.00	0.00	0.00
TOTAL	54	**65**	53	52	51.24	**64.58**	51.09	50.12
Temporal Fast Downward (IPC-2014 Version)								
MATCHCELLAR (2011)	**20**	**20**	18	0	**19.90**	**19.90**	17.92	0.00
MATCHCELLAR (2014)	**20**	**20**	**20**	0	**19.91**	**19.91**	**19.91**	0.00
TMS (2011)	0	**2**	0	0	0.00	**2.00**	0.00	0.00
TMS (2014)	0	0	0	0	0.00	0.00	0.00	0.00
TURNANDOPEN (2011)	**19**	**19**	0	0	**16.96**	16.65	0.00	0.00
TURNANDOPEN (2014)	**7**	1	0	0	**7.00**	0.86	0.00	0.00
TOTAL	**66**	62	38	0	**63.77**	59.32	37.82	0.00

Table 1: Results on Temporally Expressive Domains

lifted PDDL operators. On the other hand, disjunctive landmarks add a significant overhead to the compilation, as they not only increase the size of the state, but also the number of grounded actions in the problem, and the length of a solution to the compiled problem. However, it is always possible to simply ignore the disjunctive landmarks, and use only the single landmarks in the compilation. In the empirical evaluation, we will examine the effects of limiting the size of the disjunctions we consider.

Empirical Evaluation

Having described ways to discover and exploit temporal landmarks, in this section we examine the question of how useful they are. As a first step, we examine how temporal landmarks help different types of planners, and thus we perform an empirical evaluation of the temporal landmark compilation technique, and compare the performance of different planners on the original problem, and on compiled versions of the same problem, enriched with landmark knowledge. While we would have liked to compare our approach with that of Marzal, Sebastia, and Onaindia (2014), there was no readily available implementation of their technique.

We implemented our landmark discovery algorithm on top of OPTIC (Benton, Coles, and Coles 2012). We used the goal propositions as the initial landmarks for backchaining, and stopped backchaining as soon as a known landmark was derived. Preliminary experiments comparing the use of this approach vs. stopping backchaining only when a duplicate landmark is encountered in the same chain showed that a makespan estimate from the initial state was almost always the same when the TRPG was also used to refine temporal bounds. However, our chosen technique is much faster.

We used three different planners in our experiments: POPF (Coles et al. 2010; 2011), Temporal Fast Downward (Eyerich, Mattmüller, and Röger 2009), and YAHSP (Vidal 2004). Specifically, we used the IPC-2011 version of POPF, the IPC-2014 version of Temporal Fast Downward, and the YAHSP3-MT variant of YAHSP from IPC-2014. These are all either winners or runners-up in the temporal satisficing tracks of IPC-2011 and IPC-2014.

The planners we chose represent different types of planners: POPF is a temporally expressive planner which performs a lot of temporal reasoning, Temporal Fast Downward is also temporally expressive, but performs rather limited temporal reasoning, and YAHSP all but ignores the temporal aspect of the problem, and attempts to find a sequence of actions. We ran these planners on the problems from the temporal satisficing track of IPC-2011 (García-Olaya, Jiménez, and Linares López 2011) and IPC-2014, using the automated tools that ran IPC-2011(Linares López, Jiménez, and Helmert 2013).

For each problem instance, we created three compiled versions: one enriched only with non-disjunctive landmarks (e1), one enriched with landmarks with disjunctions up to size 4 (e4), and one enriched with all landmarks (e∞). In our evaluation, we simulated a planner with a 30 minute time limit, which spends 5 minutes on temporal landmark discovery. Landmark discovery was performed on the development machine, once for each problem instance, and was limited to

Domain	Solved				IPC Score			
	orig	e1	e4	e∞	orig	e1	e4	e∞
POPF2 (IPC-2011 Version)								
CREWPLANNING (2011)	**20**	**20**	16	16	**20.00**	**20.00**	16.00	16.00
DRIVERLOG (2014)	0	0	0	0	0.00	0.00	0.00	0.00
ELEVATORS (2011)	**3**	0	1	0	**2.18**	0.00	0.45	0.00
FLOORTILE (2011)	1	0	**2**	**2**	0.89	0.00	**1.21**	**1.21**
FLOORTILE (2014)	0	0	0	0	0.00	0.00	0.00	0.00
MAPANALYSER (2014)	0	0	0	0	0.00	0.00	0.00	0.00
OPENSTACKS (2011)	20	20	20	20	16.59	16.59	16.59	16.59
PARCPRINTER (2011)	0	0	1	**5**	0.00	0.00	1.00	**4.48**
PARKING (2011)	**20**	19	19	18	**17.42**	16.29	16.29	13.29
PARKING (2014)	12	12	12	**17**	9.16	9.89	9.89	**11.31**
PEGSOL (2011)	**19**	**19**	10	3	**18.77**	**18.77**	9.47	2.89
RTAM (2014)	0	0	0	0	0.00	0.00	0.00	0.00
SATELLITE (2014)	**4**	**4**	2	2	3.67	**3.79**	1.81	1.92
SOKOBAN (2011)	**3**	**3**	2	0	2.54	**2.67**	1.79	0.00
STORAGE (2011)	0	0	0	0	0.00	0.00	0.00	0.00
STORAGE (2014)	0	0	0	0	0.00	0.00	0.00	0.00
TOTAL	**102**	97	85	83	**91.22**	88.00	74.50	67.68
Temporal Fast Downard (IPC-2014 Version)								
CREWPLANNING (2011)	**20**	**20**	6	6	**19.85**	19.84	6.00	6.00
DRIVERLOG (2014)	0	0	0	0	0.00	0.00	0.00	0.00
ELEVATORS (2011)	**20**	19	5	6	**19.03**	18.04	4.95	5.95
FLOORTILE (2011)	**5**	**5**	0	0	**5.00**	4.44	0.00	0.00
FLOORTILE (2014)	0	0	0	0	0.00	0.00	0.00	0.00
MAPANALYSER (2014)	**17**	**17**	0	0	15.58	**16.20**	0.00	0.00
OPENSTACKS (2011)	**20**	**20**	**20**	0	**19.84**	18.99	18.99	0.00
PARCPRINTER (2011)	**10**	0	0	0	**9.88**	0.00	0.00	0.00
PARKING (2011)	**20**	10	10	0	**19.14**	6.67	6.67	0.00
PARKING (2014)	**20**	**20**	19	0	**16.18**	14.25	13.75	0.00
PEGSOL (2011)	**19**	**19**	0	0	**18.42**	18.18	0.00	0.00
RTAM (2014)	0	0	0	0	0.00	0.00	0.00	0.00
SATELLITE (2014)	**17**	8	1	0	**12.57**	4.87	0.46	0.00
SOKOBAN (2011)	**5**	1	0	0	**4.94**	0.78	0.00	0.00
STORAGE (2011)	0	0	0	0	0.00	0.00	0.00	0.00
STORAGE (2014)	0	0	0	0	0.00	0.00	0.00	0.00
TOTAL	**173**	139	61	12	**160.44**	122.25	50.82	11.95
YAHSP3-MT (IPC-2014 Version)								
CREWPLANNING (2011)	20	20	20	20	**19.88**	18.54	17.30	17.31
DRIVERLOG (2014)	**3**	**3**	0	2	1.77	**1.86**	0.00	1.57
ELEVATORS (2011)	**20**	10	9	8	**12.37**	5.39	4.65	4.27
FLOORTILE (2011)	**11**	10	2	2	**9.29**	8.43	1.10	1.10
FLOORTILE (2014)	**6**	5	1	0	**5.83**	4.88	1.00	0.00
MAPANALYSER (2014)	1	1	1	1	0.96	0.89	**0.98**	0.85
OPENSTACKS (2011)	20	20	20	20	**14.47**	13.61	13.61	13.66
PARCPRINTER (2011)	1	3	**5**	3	1.00	2.27	**2.65**	1.38
PARKING (2011)	**20**	**20**	18	15	**15.74**	14.68	13.39	9.47
PARKING (2014)	20	20	20	20	17.96	**19.27**	19.09	16.85
PEGSOL (2011)	**20**	**20**	17	13	18.52	**19.14**	15.97	11.80
RTAM (2014)	0	0	0	0	0.00	0.00	0.00	0.00
SATELLITE (2014)	20	20	20	20	17.46	**18.06**	16.71	13.48
SOKOBAN (2011)	**10**	5	6	1	**8.69**	4.34	4.33	0.48
STORAGE (2011)	7	**8**	7	0	**6.52**	4.46	3.91	0.00
STORAGE (2014)	**9**	**9**	4	0	**8.41**	5.35	1.93	0.00
TOTAL	**188**	174	150	125	**158.87**	141.17	116.61	92.21

Table 2: Results on Non Temporally Expressive Domains

		POPF				TFD			
IPC	#	orig	e1	e4	e∞	orig	e1	e4	e∞
2011	1	7	4	27	28	—	273	—	—
2011	2	12	7	11	13	—	411	—	—
2011	3	—	228	—	—	—	—	—	—
2011	13	20	10	14	21	—	—	—	—
2011	14	31	17	21	24	—	—	—	—
2011	15	49	28	—	—	—	—	—	—
2011	16	—	41	—	—	—	—	—	—
2011	17	—	61	—	—	—	—	—	—
2011	18	—	86	—	—	—	—	—	—
2011	19	—	118	—	—	—	—	—	—
2011	20	—	174	—	—	—	—	—	—
2014	1	—	41	—	—	—	—	—	—
2014	12	—	72	—	—	—	—	—	—
2014	14	—	98	—	—	—	—	—	—
2014	15	—	151	—	—	—	—	—	—
2014	16	—	189	—	—	—	—	—	—
2014	17	—	235	—	—	—	—	—	—

(a) TMS

		POPF				TFD			
IPC	#	orig	e1	e4	e∞	orig	e1	e4	e∞
2011	1	1	1	4	3	1	6	X	X
2011	2	2	2	2	3	5	2	X	X
2011	3	—	—	—	—	21	31	X	—
2011	4	—	—	—	—	17	100	X	—
2011	5	—	—	—	—	37	163	—	—
2011	6	—	—	—	—	22	151	—	—
2011	7	—	—	—	—	36	141	—	—
2011	8	—	—	—	—	55	88	—	—
2011	9	—	—	—	—	121	—	—	—
2011	10	—	—	—	—	—	396	—	—
2011	11	—	—	—	—	93	512	X	X
2011	12	—	—	—	—	130	485	—	—
2011	13	2	3	7	28	3	2	X	X
2011	14	6	2	9	13	2	4	X	X
2011	15	8	12	10	37	4	19	X	X
2011	16	10	13	6	10	4	5	X	X
2011	17	17	16	22	31	20	11	—	X
2011	18	43	14	43	47	10	19	X	—
2011	19	174	—	248	—	10	35	—	—
2011	20	—	—	—	—	11	X	—	—
2014	1	—	—	—	—	183	—	—	—
2014	2	—	—	—	—	136	—	—	—
2014	3	—	—	—	—	259	—	—	—
2014	4	—	—	—	—	248	702	—	—
2014	6	—	—	—	—	224	—	—	—
2014	7	—	—	—	—	457	—	—	—
2014	10	—	—	—	—	1130	—	—	—

(b) TURNANDOPEN

		POPF				TFD			
IPC	#	orig	e1	e4	e∞	orig	e1	e4	e∞
2011	1	0	0	0	0	1	1	X	X
2011	2	0	0	0	0	1	0	X	X
2011	3	0	0	0	1	2	1	2	X
2011	4	0	0	0	0	2	1	1	X
2011	5	0	0	0	1	2	3	2	X
2011	6	1	0	1	1	3	2	3	X
2011	7	0	0	0	1	2	3	3	X
2011	8	0	0	0	1	4	4	4	X
2011	9	0	0	0	1	12	6	5	X
2011	10	0	0	0	1	6	6	5	X
2011	11	0	0	0	2	6	7	7	X
2011	12	0	0	1	2	9	8	8	X
2011	13	0	0	0	0	0	1	2	X
2011	14	0	0	0	0	1	0	0	X
2011	15	0	0	0	1	21	0	1	X
2011	16	0	0	0	1	0	1	1	X
2011	17	0	1	0	1	1	1	0	X
2011	18	0	0	0	1	1	1	1	X
2011	19	0	0	0	0	1	14	1	X
2011	20	0	0	0	0	1	1	3	X
2014	1	0	0	0	0	1	1	2	X
2014	2	0	0	0	0	1	1	3	X
2014	3	0	0	0	0	1	1	2	X
2014	4	0	0	0	0	2	2	2	X
2014	5	1	0	0	0	1	2	3	X
2014	6	1	0	0	0	2	2	3	X
2014	7	0	0	0	0	2	3	4	X
2014	8	1	0	0	1	2	3	4	X
2014	9	0	0	0	1	2	4	6	X
2014	10	0	1	0	1	2	5	6	X
2014	11	0	0	1	1	3	4	6	X
2014	12	0	0	0	1	4	5	6	X
2014	13	0	0	0	1	3	7	8	X
2014	14	0	0	0	1	3	7	7	X
2014	15	1	0	0	2	5	6	9	X
2014	16	0	0	1	1	5	8	10	X
2014	17	0	0	0	2	5	8	10	X
2014	18	1	0	0	2	23	11	12	X
2014	19	0	0	0	2	7	10	36	X
2014	20	0	0	0	2	7	11	17	X

(c) MATCHCELLAR

Table 3: Time until first solution on each instance of the problems featuring requires concurrency. Solution times are in seconds, — indicates a timeout, X indicates an invalid solution was found. Instances which were not solved at all are omitted.

5 minutes of CPU time. If landmark discovery did not terminate in 5 minutes, we used an empty set of landmarks (in which case the compiled problem was the original problem). On the compiled problems, CPU time for running the planner was limited to 25 minutes (regardless of how long landmark discovery took on that specific problem), and CPU time on the original problems was limited to 30 minutes. In both cases, the memory limit was 6 GB.

Timeouts in landmark discovery occurred in 14 out of 430 problems: 6 from ELEVATORS (2011), 6 from DRIVERLOG (2014), and 2 from TMS (2014). Of these, only the ELEVATORS (2011) instances were solved by any planner in our experiments; the others were not solved either with or without landmarks. Additionally, if the planners running on the compiled problems enriched with temporal landmarks had the full 30 minutes, Temporal Fast Downward and YAHSP would have solved 1 or 2 more problems, depending on the maximum disjunction size, while the results for POPF would have been the same.

We split our presentation of the results into domains which feature required concurrency (temporally expressive), and those which do not. Table 1 shows the number of problems solved in each domain and the IPC score for POPF and Temporal Fast Downward on the temporally expressive domains. YAHSP was omitted, as it can not solve any of these problems. Looking at these results, we can see a significant benefit from using temporal landmarks in the TMS domain. To examine this domain in more detail, Table 3a shows the solution time on each instance of TMS that was solved by any planner. These results show that multiple problems which were not solved by POPF in 1800 seconds without landmarks were solved by the same planner when enriched with the single landmarks, an order of magnitude faster. TMS is a rich, temporally expressive, domain, and therefore it is not surprising that adding more temporal reasoning helps.

Table 3b shows detailed results for TURNANDOPEN. TURNANDOPEN is a temporally expressive version of the classical GRIPPER domain, and suffers from the same problem of many symmetric solutions. Unfortunately, temporal landmarks do not break symmetries, and thus do not help in this domain. This domain illustrates an issue with Temporal Fast Downward, which does not always return correct solutions.

Finally, Table 3c shows detailed results for MATCHCELLAR, which turns out to be easy enough for both planners to solve all problems without any help. Thus, temporal landmarks can not improve the number of problems solved. Even when solution time is considered, both planners are fast enough that there is no significant difference in solution times with or without landmarks.

Turning our attention to domains without required concurrency, Table 2 shows the number of problems solved and the IPC score in each domain for POPF, Temporal Fast Downward, and YAHSP. These results reveal that there is some benefit from using temporal landmarks, even in some non temporally expressive domains. First, temporal landmarks help improve solution quality (and thus, IPC score) in several domains. However, more interestingly, disjunctive temporal landmarks help some of the planners in a few domains, even more than single landmarks. Specifically, note that disjunctive landmarks help both POPF and YAHSP in PARCPRINTER (2011), and POPF in FLOORTILE (2011) and PARKING (2014). We believe that in these domains, the disjunctive landmarks are able to capture a key piece of knowledge about the solution, which can only be represented as a disjunction. This leads us to believe that directly integrating the guidance from landmarks into the planner, without the overhead of compiling disjunctive landmarks into the problem, will lead to even greater benefit.

Conclusion

In this paper, we described a new reasoning mechanism for discovering temporal landmarks, which combine information about what must be achieved and when it must be achieved. We described a technique for discovering such temporal landmarks, and presented both a theoretical example and empirical results which show that temporal landmarks can help temporal planners, especially on problems with complex temporal interactions. In future work, we intend to use temporal landmarks directly in a planner: both by incorporating the landmark derivation rules into the tBurton planner (Wang and Williams 2015), and by using the landmarks directly inside OPTIC (Benton, Coles, and Coles 2012), thus avoiding the large overhead of compiling disjunctive landmarks into the problem.

Acknowledgements

The work was partially supported by the DARPA MRC Program, under grant number FA8650-11-C-7192, Boeing Corporation, under grant number MIT-BA-GTA-1, and ARC project DP140104219, "Robust AI Planning for Hybrid Systems". NICTA is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program. Finally, we would like to thank the anonymous reviewers for their insightful comments, and the authors of the OPTIC planner for making it publicly available.

References

Benton, J.; Coles, A. J.; and Coles, A. I. 2012. Temporal planning with preferences and time-dependent continuous costs. In *Proc. ICAPS 2012*.

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

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

Coles, A. J.; Coles, A. I.; Clark, A.; and Gilmore, S. T. 2011. Cost-sensitive concurrent planning under duration uncertainty for service level agreements. In *Proc. ICAPS 2011*.

Cushing, W.; Kambhampati, S.; and Weld, D. S. 2007. When is temporal planning really temporal? In *Proc. IJCAI 2007*.

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

Domshlak, C.; Katz, M.; and Lefler, S. 2012. Landmark-enhanced abstraction heuristics. *AIJ* 189:48–68.

Eyerich, P.; Mattmüller, R.; and Röger, G. 2009. Using the context-enhanced additive heuristic for temporal and numeric planning. In *Proc. ICAPS 2009*, 130–137.

Fox, M., and Long, D. 2003. PDDL2.1: An extension to PDDL for expressing temporal planning domains. *JAIR* 20:61–124.

García-Olaya, A.; Jiménez, S.; and Linares López, C. 2011. The 2011 international planning competition. Technical report, Universidad Carlos III de Madrid. http://hdl.handle.net/10016/11710.

Haslum, P. 2009. Admissible makespan estimates for pddl2.1 temporal planning. In *ICAPS 2009 Workshop on Heuristics for Domain-Independent Planning*.

Hoffmann, J.; Porteous, J.; and Sebastia, L. 2004. Ordered landmarks in planning. *JAIR* 22:215–278.

Hofmann, A., and Williams, B. 2006. Robust execution of temporally flexible plans for bipedal walking devices. In *Proc. ICAPS 2006*, 386–389.

Karpas, E., and Domshlak, C. 2009. Cost-optimal planning with landmarks. In *Proc. IJCAI 2009*, 1728–1733.

Keyder, E.; Richter, S.; and Helmert, M. 2010. Sound and complete landmarks for and/or graphs. In *Proc. ECAI 2010*, 335–340.

Linares López, C.; Jiménez, S.; and Helmert, M. 2013. Automating the evaluation of planning systems. *AI Communications* 26(4):331–354.

Marzal, E.; Sebastia, L.; and Onaindia, E. 2008. Detection of unsolvable temporal planning problems through the use of landmarks. In *Proc. ECAI 2008*, 919–920.

Marzal, E.; Sebastia, L.; and Onaindia, E. 2014. On the use of temporal landmarks for planning with deadlines. In *Proc. ICAPS 2014*.

Richter, S., and Westphal, M. 2010. The LAMA planner: Guiding cost-based anytime planning with landmarks. *JAIR* 39:127–177.

Sebastia, L.; Marzal, E.; and Onaindia, E. 2007. Extracting landmarks in temporal planning domains. In *IC-AI 2007*, 520–526.

Vidal, V., and Geffner, H. 2006. Branching and pruning: An optimal temporal POCL planner based on constraint programming. *AIJ* 170(3):298–335.

Vidal, V. 2004. A lookahead strategy for heuristic search planning. In *Proc. ICAPS 2004*, 150–159.

Wang, D., and Williams, B. 2015. tBurton: A divide and conquer temporal planner. In *Proc. AAAI 2015*.