Proceedings of the Thirty-Third International Conference on Automated Planning and Scheduling (ICAPS 2023)

Fixing Plans for PDDL+ Problems: Theoretical and Practical Implications

Francesco Percassi[1], **Enrico Scala**[2], **Mauro Vallati**[1]

[1] School of Computing and Engineering, University of Huddersfield, UK
[2] University of Brescia, Italy
f.percassi@hud.ac.uk, enrico.scala@unibs.it, m.vallati@hud.ac.uk

Abstract

The plan, execution, and replan framework has proven to be extremely valuable in complex real-world applications, where the dynamics of the environment cannot be fully encoded in the domain model. However, this comes at the cost of regenerating plans from scratch, which can be expensive when expressive formalisms like PDDL+ are used. Given the complexity of generating PDDL+ plans, it would be ideal to reuse as much as possible of an existing plan, rather than generating a new one from scratch every time. To support more effective exploitation of the plan, execution, and replan framework in PDDL+, in this paper, we introduce the problem of discretised PDDL+ plan fixing, which allows one to fix existing plans according to some defined constraints. We demonstrate the theoretical implications of the introduced notion and introduce reformulations to address the problem using domain-independent planning engines. Our results show that such reformulations can outperform replanning from scratch and unlock planning engines to solve more problems with fine-grained discretisations.

Introduction

A planning knowledge model is intrinsically an approximation of the dynamics of a corresponding application domain (McCluskey and Porteous 1997). On the one hand, this is needed to allow planning engines to generate solution plans within some time limit. On the other hand, such an approximation can lead to plans that are not applicable in the real-world due to some dynamics that are not fully modelled. The latter aspect is exacerbated in hybrid discrete-continuous models represented using PDDL+ (Fox and Long 2006), where the notoriously challenging nature of the problems forces knowledge engineers to significantly simplify the models to enable the exploitation of planning in complex real-world scenarios. To ensure the suitability of the generated solutions for the target application, a traditional framework is based on a continual planning loop (desJardins et al. 1999; Brenner and Nebel 2009) that involves the steps of plan, execution and replan. In other words, the execution of a generated plan is monitored to identify discrepancies from the predicted behaviour and, if that is the case, a brand new solution plan is generated based on some updated knowledge. While this framework can be very beneficial in some situations, it can show its limits in complex scenarios and some form of plan reuse is usually much more beneficial (van der Krogt and de Weerdt 2005; Fox et al. 2006; Arangú, Garrido, and Onaindia 2008; Gerevini and Serina 2010; Scala 2014; Scala and Torasso 2015). This is even more critical in PDDL+, due to the complexity of the plan generation process that needs to take into account discrete and continuous numeric and temporal aspects. In PDDL+, it would be ideal to reuse as much as possible of an existing plan, rather than generate a new one from scratch. To nurture this idea, in this paper we introduce the notion of PDDL+ plan fixing in the context where PDDL+ is interpreted under discrete semantics (Penna, Magazzeni, and Mercorio 2012; Piotrowski et al. 2016; Percassi, Scala, and Vallati 2023). Our aim is that of providing an approach to fix existing plans according to some defined constraints. The notion of plan fixing that we propose is overarching and covers a spectrum of problems that range from planning from scratch to validation.

Unlike previous findings in classical planning (Nebel and Koehler 1995), which demonstrate that repairing a plan can be more challenging than creating a new one from scratch, our paper identifies a specific subset of plan repair problems that are provably simpler than replanning. This subset of problems is defined by the constraint that the repaired plan must only contain actions from the original invalid plan, albeit rescheduled over time. This subset of problems can be proven to be NP-COMPLETE, while a replanning from scratch would induce an undecidable problem.

The main contributions of this work are as follows: (i) we introduce and characterise a spectrum of PDDL+ plan fixing problems, and provide a computational complexity analysis; (ii) we observe that starting from plan validation as planning, it is possible to define a set of simple PDDL+ reformulations that allow one to encode the plan fixing problem in PDDL+, hence supporting the use of domain-independent planning engines to effectively tackle it; and (iii), we provide an extensive experimental analysis assessing the value of the plan fixing problem; our results indicate that not only plan fixing can speed up substantially the planning process but can unlock the planning engine to find plans for much finer discretisation inputs.

Background

This section reports on the PDDL+ problem (Fox and Long 2006) interpreted over a discrete timeline (Percassi, Scala, and Vallati 2023), which is the focus of this work.

A PDDL+ planning problem Π is a tuple $\langle F, X, I, G, A, E, P\rangle$ in which F and X are the sets of Boolean and numeric variables taking values from $\{\top, \bot\}$ and $\mathbb{Q}$, respectively, that can participate in the definition of propositional formulas having numeric and Boolean conditions as terms. More specifically, a numeric condition is of the form $\langle \xi \bowtie 0\rangle$ with ξ being a numeric expression over X and $\mathbb{Q}$, and $\bowtie \in \{\leq, <, =, >, \geq\}$. A Boolean condition is of the form $\langle f = \{\top, \bot\}\rangle$ with f being a Boolean variable in F. A formula is constructed by combining Boolean and numeric conditions using standard logical connectives. I is the description of the initial state expressed as a full assignment to all variables in X and F. G is the description of the goal, expressed as a formula. A and E are the sets of actions and events, respectively. Actions and events are pairs $\langle p, e\rangle$ where p is a formula and e is a set of Boolean and numeric assignments. Boolean assignments have the form $\langle f := \{\bot, \top\}\rangle$ while numeric assignments have the form $\langle \{asgn, inc, dec\}, x, \xi\rangle$, where $x \in X$ and ξ is a numeric expression over X and $\mathbb{Q}$. P is a set of processes. A process is defined as a pair $\langle p, e'\rangle$, where p is a formula and e' is a set of continuous numeric effects expressed as pairs $\langle x, \xi\rangle$, where ξ represents the additive contribution to the derivative of x as time progresses. Under discrete semantics, this continuous change is discretised using the function $\Delta(\xi, \delta) = \xi \cdot \delta$.

Let $a = \langle p, e\rangle$ be an action/event/process, we use *pre*(a) to refer to the precondition p of a, and *eff*(a) to the effect e of a. Moreover, in the following, we will use a, ρ, and ε to refer to a generic action, process, and event, respectively. In order to make the notation more concise, Boolean conditions (assignments) of the form $\langle f = \top\rangle$ ($\langle f := \top\rangle$) and $\langle f = \bot\rangle$ ($\langle f := \bot\rangle$) are shortened to f and $\neg f$, respectively.

A PDDL+ plan π_t is a pair $\langle \pi, \langle t_s, t_e\rangle\rangle$ where $\pi = \langle\langle a_1, t_1\rangle, ..., \langle a_n, t_n\rangle\rangle$ is a sequence of timestamped actions, with $t_i \in \mathbb{Q}$. The pair $\langle t_s, t_e\rangle$, with $t_s, t_e \in \mathbb{Q}$ and $t_s \leq t_e$, represents the envelope in which the plan π is executed. The subscript i in $\langle a_i, t_i\rangle$ denotes the i-th action in the sequence. We say that a PDDL+ plan π_t is well-formed if for all $i, j \in \{1, ..., n\}$ with $i < j$, the conditions $t_i \leq t_j$ and $t_s \leq t_i \leq t_e$ hold. From now on, we will only consider well-formed plans.

Given a set S, we use $seq(S)$ to indicate an arbitrary ordering of its elements. With a slight abuse of notation, we use the symbol $\in$ to indicate that an element belongs to a sequence of elements, i.e., $s \in seq(S)$.

Building on our previous work (Percassi, Scala, and Vallati 2023) and the work of Shin and Davis (2005), we formalise the semantics of PDDL+ by introducing the concepts of time points, histories, and discrete plan projection. In the following discussion, we assume that the reader is familiar with the well-known notions of action/event applicability. We use the notation $\gamma(s, \cdot)$ to denote the state resulting from the application of either an action/event ($\gamma(s, a)$) or a sequence of actions/events ($\gamma(s, \langle a_1, ..., a_n\rangle)$) in a state s.

Given a discretisation step $\delta \in \mathbb{Q}_{>0}$, a time point T is a pair $\langle t = \delta \cdot n, n'\rangle$ where $n \in \mathbb{Z}$ and $n' \in \mathbb{N}$; t denotes the clock of T while n' is the counter used to order actions and events happening at t. Time points over $\mathbb{Q} \times \mathbb{N}$ are ordered lexicographically. A history $\mathbb{H}$ over an interval $\mathcal{I} = [T_s, T_e]$ maps each time point in $\mathcal{I}$ into a situation. A "situation at time point T" is the tuple $\mathbb{H}(T) = \langle \mathbb{H}_A(T), \mathbb{H}_s(T)\rangle$, where $\mathbb{H}_A(T)$ is the action executed at time point T and $\mathbb{H}_s(T)$ is a state, i.e., an assignment to all variables in X and F at time point T. We denote by $\mathbb{H}_s(T)[v]$ and $\mathbb{H}_s(T)[\xi]$ the value assumed by $v \in F \cup X$ and by a numeric expression ξ, respectively, in state s at time T. $E_{trigg}(T)$ indicates the sequence of triggered events in T. T is a significant time point (STP) of $\mathbb{H}$ over $\mathcal{I}$ iff either at least an action is applied ($\mathbb{H}_A(T) \neq \langle\rangle$), a non-empty sequence of events is triggered ($E_{trigg}(T) \neq \langle\rangle$), at least a process ρ has started or stopped, i.e., the precondition of ρ becomes (un)satisfied, or there has been a discrete change just before. Following Fox and Long (2006) we assume PDDL+ problems *event-deterministic* (given a state in which multiple events are triggered, we can sequence them arbitrarily always obtaining the same outcome) and *finite complexity*, i.e., the PDDL+ problem Π has to induce a finite number of spontaneous changes over an interval. For events, this requires that cascades of events have to be finite, which can be achieved by imposing that an event must be self-deactivating and that, given a timestamp, an event can be triggered at most once (Fox, Howey, and Long 2005).

To formalise plan fixing and validation, we need to define the definition of (discrete) PDDL+ plan projection. The plan projection is defined by four rules. The first two rules (R1–R2) state that when an action or an event occurs at a specific time point, there must be a successor state with the same clock time and an increased counter. The third rule (R3) ensures that the actions in the original PDDL+ plan are projected while maintaining their original order. The fourth rule (R4) describes how numeric variables change in a discrete manner when time advances by a discrete quantity δ.

Definition 1 (Discrete PDDL+ Plan Projection)**.** *Let $\delta \in \mathbb{Q}_{>0}$, $\mathbb{H}$ be a history, I be an initial state and π_t be a* PDDL+ *plan. We say that $\mathbb{H}$ is a* discrete projection of π_t which starts in I *iff $\mathbb{H}$ induces the STPs $T_{\mathbb{H}} = \langle T_0 = \langle t_s, 0\rangle, ..., T_m = \langle t_e, n_m\rangle\rangle$ where either $t_{i+1} = t_i + \delta$ or $t_{i+1} = t_i$ and, for all $i \in \{0, ..., m\}$, the following rules hold:*

- **R1** *$E_{trigg}(T_i) \neq \langle\rangle$ iff $\mathbb{H}_s(T_{i+1}) = \gamma(\mathbb{H}_s(T_i), E_{trigg}(T_i))$, $\mathbb{H}_A(T_i) = \langle\rangle$, $t_{i+1} = t_i$ and $n_{i+1} = n_i + 1$;*
- **R2** *$\mathbb{H}_A(T_i) \neq \langle\rangle$ iff $\mathbb{H}_s(T_{i+1}) = \gamma(\mathbb{H}_s(T_i), \mathbb{H}_A(T_i))$, $E_{trigg}(T_i) = \langle\rangle$, $t_{i+1} = t_i$ and $n_{i+1} = n_i + 1$;*
- **R3** *for each $\langle a_i, t_i\rangle, \langle a_j, t_j\rangle$ in π, with $i < j$ and $t_i = t_j$ there exists T_k, T_z in $T_{\mathbb{H}}$ such that a_i in $\mathbb{H}_A(T_k)$ and a_j in $\mathbb{H}_A(T_z)$ where $t_k = t_z = t_i$ and $n_k < n_z$;*
- **R4** *for each pair of contiguous STPs $T_i = \langle t_i, n_i\rangle$, $T_{i+1} = \langle t_{i+1}, 0\rangle$ such that $t_{i+1} = t_i + \delta$, the value of each numeric variable $x \in X$ is updated as:*

$$\mathbb{H}_s(T_{i+1})[x] = \mathbb{H}_s(T_i)[x] + \sum_{\substack{\langle x', \xi\rangle \in \mathit{eff}(\rho),\ x' = x \\ \rho \in P \text{ such that } \mathbb{H}_s(T_i) \models \mathit{pre}(\rho)}} \mathbb{H}_s(T_i)[\Delta(\xi, \delta)]$$

After defining the plan projection, we can determine the validity of a plan by checking that every action is applicable in the corresponding state and that the plan results in the goal in the final state.

Definition 2 (Valid PDDL+ Plan Under δ Discretisation)**.** *π_t is a* valid PDDL+ plan for Π under δ discretisation *iff $\mathbb{H}_s(T_m) \models G$ and, for each $T \in \mathcal{I}$ such that $\mathbb{H}_A(T) \neq \langle\rangle$, then $\mathbb{H}_A(T)$ is applicable.*

Definition 3 (Discrete Validation Problem)**.** *A* discrete validation problem *(shortened as* VALIDATION*) is a tuple $\langle \Pi, \pi_t \rangle$ where Π is a* PDDL+ *problem and π_t is a plan for Π.* VALIDATION *aims at evaluating whether π_t is valid for Π under δ discretisation.*

Validation as Planning

In our previous work (Percassi, Scala, and Vallati 2022) we addressed the VALIDATION problem as a PDDL+ planning problem. Specifically, given a PDDL+ problem and a plan to be validated, we proposed a collection of reformulations to generate a novel PDDL+ planning problem, called the validating problem, that admits a solution iff the considered plan is valid. We briefly summarise the baseline reformulation, namely $\mathcal{V}_0$, because it is a key element for how we deal with fixing PDDL+ plans.

Let $\Pi = \langle F, X, I, G, A, E, P \rangle$ be a PDDL+ problem, and let $\pi_t = \langle \pi, \langle t_s, t_e \rangle \rangle$ be a plan for Π. $\mathcal{V}_0$ produces a new validating PDDL+ problem $\Pi^{\pi_t}_{\mathcal{V}_0} = \langle F \cup F_A \cup \{\text{T}\}, X \cup \{\mathit{time}\}, I \cup \{\mathit{a\text{-}d}_0, \langle \mathit{time} := t_s \rangle, \text{T}\}, G \wedge \mathit{a\text{-}d}_n \wedge \langle \mathit{time} = t_e \rangle, A_\pi, E, P \cup \{\rho_{\mathit{time}}\} \rangle$ such that $\rho_{\mathit{time}} = \langle \text{T}, \{\langle \mathit{time}, 1 \rangle\} \rangle$ and:

$$F_A = \{\mathit{a\text{-}d}_i \mid i \in \{1, ..., n = |\pi|\}\}$$

$$A_\pi = \bigcup_{\langle a_i, t_i \rangle \in \pi} \{\langle \mathit{pre}(a_i) \wedge \mathit{a\text{-}d}_{i-1} \wedge \neg \mathit{a\text{-}d}_i \wedge \langle \mathit{time} = t_i \rangle, \mathit{eff}(a_i) \cup \{\mathit{a\text{-}d}_i\} \rangle\}$$

Essentially, $\mathcal{V}_0$ induces a problem $\Pi^{\pi_t}_{\mathcal{V}_0}$ in which the only executable actions are those belonging to the plan to be validated. Such actions are appropriately modified to be executed in the same ordering in which they appear in the plan to be validated and at the same timestamps. The additional predicates F_A are used for ensuring the ordering preservation; specifically, each predicate $\mathit{a\text{-}d}_i \in F_A$ tracks whether the action instance a_i of π_t has been executed. Additionally, the goal must be achieved within the same envelope.

Plan Fixing Problems and Properties

We are interested in finding ways to effectively fix an invalid PDDL+ plan to avoid replanning from scratch. To this end, we formally introduce the *discrete plan fix problem*. Intuitively, given a PDDL+ problem and a plan π_t eventually invalid for Π, a discrete plan fix problem aims at finding a fixed plan π'_t obtained by using a limited set of manipulations over π_t. The possible manipulations are modelled by a set of constraints $\mathcal{C}$; these constraints implicitly define the space of possible fixed plans. We refer to this problem FIXABILITY as the aim consists in evaluating whether the input plan is *fixable* according to the given constraints.

In the following, we provide the formal and general definition of FIXABILITY and then provide details on some specialisations for the constraint set $\mathcal{C}$. We focus on specialisations that can lead to problems that can be dealt with more efficiently than replanning from scratch, both in theory and in practice, while still retaining a reasonable degree of flexibility.

Definition 4 (Discrete Plan Fix Problem)**.** *A* discrete plan fix problem *(shortened as* FIXABILITY*) is a tuple $\langle \Pi, \pi_t, \mathcal{C} \rangle$ where Π is a* PDDL+ *problem, $\pi_t = \langle \pi, \langle t_s, t_e \rangle \rangle$ is an (eventually) invalid plan for Π and $\mathcal{C}$ is a set of constraints. The* FIXABILITY problem *aims at generating a valid plan $\pi'_t = \langle \pi', \langle t_s, t'_e \rangle \rangle$ for Π such that π'_t satisfies the constraints $\mathcal{C}$ w.r.t. π_t.*

Now, we will detail qualitatively the language we are considering for specifying the set of constraints $\mathcal{C}$. We will do this by using different primitives: to express that an action a_i in π has to appear in π', without predicating about its position and timestamp, we use $in(a_i, \pi')$; to express that an action a_i has to be placed in the j-th position of π', we use $pos(a_i, j, \pi')$; to express that an action has to be executed within a specific time window, we write $time(a_i, [l, u], \pi')$ where $l, u \in \mathbb{Q}$ and $l \leq u$; to bound from above the envelope of the fixed plan we use $\langle t'_e \leq t_e + \sigma \rangle$ where $\sigma \in \mathbb{Q}_{\geq 0}$; finally, we can require that π_t and π'_t have the same envelope, i.e., $\langle t'_e = t_e \rangle$. Let $\langle a_i, t_i \rangle$ be the i-th action in π, we denote with $\mathcal{TW}(t_i, \omega) = [\mathcal{TW}_l(t_i, \omega), \mathcal{TW}_u(t_i, \omega)] = [\max(t_s, t_i - \frac{\omega}{2}), t_i + \frac{\omega}{2}]$ the time window around t_i wide as $\omega \in \mathbb{Q}_{\geq 0}$.

The simplest specialisation is when $\mathcal{C} = \emptyset$; in this case, we indicate the fixing problem with FIXABILITY$_\emptyset$. Then we have FIXABILITY$_\mathcal{I}$ ($\mathcal{I}$ stands for *inclusion*), which requires that the only actions that have to appear in π' are exclusively those that appear in π but does not prescribe any timestamps nor ordering. FIXABILITY$_\mathcal{S}$ ($\mathcal{S}$ stands for *sorting*) requires that each application of an action in π has to appear in π' by preserving the overall ordering of π. FIXABILITY$_\mathcal{W}$ ($\mathcal{W}$ stands for *time window*) requires that each application of an action in π has to appear in π' in a specific time window. Finally, FIXABILITY$_{\mathcal{WS}}$ requires that π_t has to jointly comply with the constraints imposed by FIXABILITY$_\mathcal{W}$ and FIXABILITY$_\mathcal{S}$. Additionally, given $Z \in \{\mathcal{I}, \mathcal{S}, \mathcal{W}, \mathcal{WS}\}$, we introduce a *temporally bounded* variant FIXABILITY$_Z$ for each of the previously described specialisations. In such a variant FIXABILITY$^{\leq}_Z$, the length of the repaired plan is restricted so as not to exceed a threshold determined by $\sigma \in \mathbb{Q}_{\geq 0}$. Specifically, we require that $t'_e \leq t_e + \sigma$. More formally:

Definition 5 (FIXABILITY Specialisation)**.** *Given an instance of the* FIXABILITY *problem $\langle \Pi, \pi_t, \mathcal{C} \rangle$, we define the following specialisation based on the definition of $\mathcal{C}$:*

- FIXABILITY$_\emptyset$*, if $\mathcal{C} = \emptyset$;*
- FIXABILITY$_\mathcal{I}$ *if $\mathcal{C} = \mathcal{C}_\mathcal{I} = \mathcal{C}_\in \cup \mathcal{C}_\notin$ where:*
  $$\mathcal{C}_\in = \{\langle in(a_i, \pi') \rangle \mid \langle a_i, t_i \rangle \in \pi\}$$
  $$\mathcal{C}_\notin = \{\neg\langle in(a, \pi') \rangle \mid a \in A \wedge a \notin \{a \mid \langle a, t \rangle \in \pi\}\}$$
- FIXABILITY$_\mathcal{S}$*, if $\mathcal{C} = \mathcal{C}_\mathcal{I} \cup \mathcal{C}_\mathcal{S}$:*
  $$\mathcal{C}_\mathcal{S} = \{pos(a_i, i, \pi') \mid \langle a_i, t_i \rangle \in \pi\}$$

- FIXABILITY$_{\mathcal{W}}$, *if* $\mathcal{C} = \mathcal{C}_{\mathcal{I}} \cup \mathcal{C}_{\mathcal{W}}(\omega)$ *where* $\omega \in \mathbb{Q}_{\geq 0}$ *and:*

$$\mathcal{C}_{\mathcal{W}}(\omega) = \{\langle time(a_i, \mathcal{TW}(t_i, \omega), \pi' \rangle \mid \langle a_i, t_i \rangle \in \pi\}$$

- FIXABILITY$_{\mathcal{WS}}$ *when* $\mathcal{C} = \mathcal{C}_{\mathcal{I}} \cup \mathcal{C}_{\mathcal{W}}(\omega) \cup \mathcal{C}_{\mathcal{S}}$.
- FIXABILITY$^{\leq}_{Z}$, *if* $\mathcal{C} = \mathcal{C}_Z \cup \{\langle t'_e \leq t_e + \sigma \rangle\}$ *where* $\mathcal{C}_Z$ *is the set of constraints imposed by* FIXABILITY$_Z$ *with* $Z \in \{\mathcal{I}, \mathcal{S}, \mathcal{W}, \mathcal{WS}\}$, *and* $\sigma \in \mathbb{Q}_{\geq 0}$.

Note that, FIXABILITY$_{\emptyset}$ corresponds to a PDDL+ problem as it does not impose any constraints on the repaired plan. On the other extreme, the VALIDATION problem can be reformulated as a highly constrained FIXABILITY problem, i.e., $\langle \Pi, \pi_t, \mathcal{C} \rangle$ where $\mathcal{C} = \mathcal{C}_{\mathcal{I}} \cup \mathcal{C}_{\mathcal{S}} \cup \mathcal{C}_{\mathcal{W}}(\omega = 0) \cup \{\langle t'_e = t_e \rangle\}$. Note that in order to ensure that the actions of π'_t are executed at their original timestamps, the set of constraints $\mathcal{C}_{\mathcal{W}}(\omega = 0)$ imposes a time window of width zero for each timestamped action of π.

We remark that the constraints $\mathcal{C}$ have to be intended as hard constraints. Then, the constraints expressed under the form of $pos(\cdot)$ and $time(\cdot)$ imply those expressed by $in(\cdot)$; e.g., the constraint $\langle pos(a_i, j, \pi') \rangle$, which requires that a_i is the j-th action of π', implies that a_i has to be necessarily in π' and therefore, when $\langle pos(a_i, j, \pi') \rangle$ holds $in(a_i, \pi')$ holds as well.

Definition 5 induces a hierarchical relation among different FIXABILITY problems (Figure 1). To be specific, given two plan fixing problems FIXABILITY and FIXABILITY$'$ with constraints $\mathcal{C}$ and $\mathcal{C}'$, we say that FIXABILITY is a restriction of FIXABILITY$'$, and we write this relationship as FIXABILITY $\prec_{\text{restrict}}$ FIXABILITY$'$, iff $\mathcal{C}' \subset \mathcal{C}$. It is easy to see that the solution space of FIXABILITY is a subset of the one of FIXABILITY$'$. Observe also that some restrictions in Figure 1 can be reconstructed transitively, e.g., FIXABILITY$_{\mathcal{W}} \prec_{\text{restrict}}$ FIXABILITY$_{\emptyset}$, which is indicated by a dotted line.

In the following, we provide some complexity results for these problems. To do so, we first need to show that VALIDATION belongs to P.

Lemma 1. VALIDATION $\in$ P.

Proof Sketch. According to Definition 2, one can validate a plan π_t for Π by generating the discrete PDDL+ plan projection $\mathbb{H}$ of π_t, check that each action is applicable and finally check that the goal is achieved in the final state. So in order to show that VALIDATION is polynomial, the somehow difficult bit is to show that $\mathbb{H}$ can be generated in polynomial time.

By Definition 1, it can be noticed that the number of generated states depends on two kinds of transitions, the *instantaneous* and *temporal* ones. Instantaneous transitions occur when an action is performed while temporal transitions occur when time advances by the discrete quantity δ. For a temporal transition, we need to apply R4 of Definition 1, and it is easy to see that this only requires a number of operations that is linear on the input. For an instantaneous transition, we need to apply function $\gamma(s, a)$ that is polynomial as well. Furthermore, every time an action is applied or time elapses by δ, we need to check the possible triggering of events. Given a state s and a set of events, we recursively apply all events that are triggered in s. Note that a state can trigger a cascade of events but since we are assuming that an event can be only executed once for each timestamp, it follows that this procedure can be called at most $|E|$ times for each transition. Therefore, in the worst case, the number of states generated is equal to $|\tau| = |E| \cdot (|\pi| + \frac{t_e - t_s}{\delta})$. □

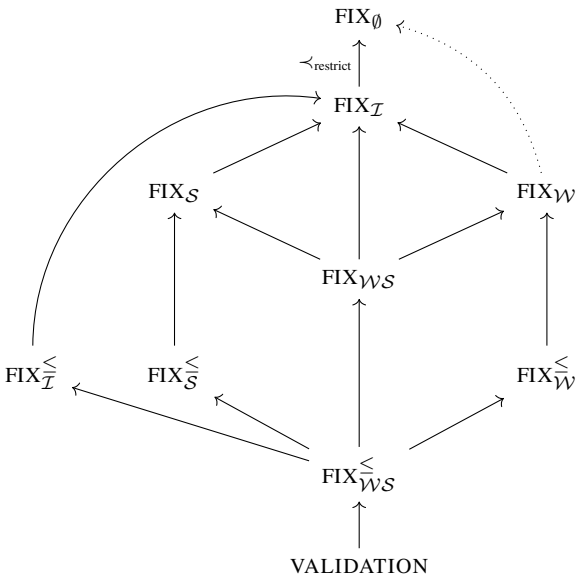

Figure 1: Hierarchy of the different FIXABILITY problems (shortened in FIX). Each arrow represents the restriction relationship between two FIXABILITY problems.

Lemma 2. FIXABILITY$^{\leq}_{\mathcal{WS}}$ *is* NP-HARD.

Proof Sketch. Let $\phi = \bigwedge_{i=1}^{n} \bigvee_{j=1}^{m} l_{ij}$ be a propositional formula and let $\mathbb{L}_\phi = \bigcup_{i=1}^{n} \bigcup_{j=1}^{m} \{l_{ij}\}$ be the set of literals involved in ϕ. We define the set of $\kappa \leq n \cdot m$ Boolean variables as $\mathbb{V}_\phi = \{\theta(l) \mid l \in \mathbb{L}_\phi\}$ where $\theta(l) = f$ if $l = \neg f$, f otherwise. Furthermore, we define an arbitrary ordering of $\mathbb{V}_\phi$, i.e., $\overline{\mathbb{V}}_\phi = seq(\mathbb{V}_\phi) = \langle f_1, ..., f_\kappa \rangle$.

The Boolean satisfiability problem of ϕ (shortened as SAT) can be polynomially reduced into a FIXABILITY$^{\leq}_{\mathcal{WS}}$ problem $\langle \Pi_\phi, \pi_t = \langle \pi, \langle 0, 2\kappa \rangle \rangle, \mathcal{C}_\phi \rangle$ in which each element is detailed as follows. The PDDL+ problem Π_ϕ is defined as $\Pi_\phi = \langle F_\phi \cup S_\phi \cup \{\top\}, \{x\}, \{\langle x := 0 \rangle, \top\}, \phi \wedge \top, A_\phi, E_\phi \cup E_\perp, \{\rho_x\} \rangle$ where $F_\phi = \{f_i \mid f_i \in \overline{\mathbb{V}}_\phi\}$, $S_\phi = \{s_{f_i} \mid f_i \in \overline{\mathbb{V}}_\phi\}$, $A_\phi = \{a_{f_i} \mid f_i \in \overline{\mathbb{V}}_\phi\}$, $E_\phi = \{\varepsilon^+_{f_i}, \varepsilon^-_{f_i} \mid f_i \in \overline{\mathbb{V}}_\phi\}$, $E_\perp = \{\langle \langle x = 2i - 1 \rangle, \{\neg\top\} \rangle \mid i \in \{1, ..., \kappa\}\}$ and $\rho_x = \langle \top, \{\langle x, 1 \rangle\} \rangle$.

Π_ϕ has the same Boolean variables of the formula ϕ, i.e., F_ϕ, augmented with the set of variables S_ϕ, a single numeric variable x, which is increased linearly by means of the process ρ_x. The set of actions is defined by mapping each (indexed) Boolean variable f_i into an action $a_{f_i} \in A_\phi$. The problem consists of placing the actions A_ϕ at the correct timestamps so that the formula ϕ is satisfied at the end of the plan. Specifically, these actions can affect indirectly the

truth value of the variable F_ϕ by the application of the sequence $\langle a_{f_i}, \varepsilon_{f_i} \rangle$, where $\varepsilon_{f_i} \in \{\varepsilon^+_{f_i}, \varepsilon^-_{f_i}\}$. Such elements are defined as:

$$a_{f_i} = \langle \top, \{s_{f_i}\} \rangle$$
$$pre(\varepsilon^+_{f_i}) = \langle s_{f_i} \wedge \langle x > 2i-1 \rangle, \{\neg s_{f_i}, f_i\} \rangle$$
$$pre(\varepsilon^-_{f_i}) = \langle s_{f_i} \wedge \langle x < 2i-1 \rangle, \{\neg s_{f_i}, \neg f_i\} \rangle$$

The execution of action a_{f_i} when $x \neq 2i-1$ immediately triggers, in mutual exclusion, one of the two associated events. The selected event assigns a truth value to f_i depending on the value of x. Once the truth value of f_i is assigned through these events, it can no longer be changed since, by negating s_{f_i}, the events $\{\varepsilon^-_{f_i}, \varepsilon^+_{f_i}\}$ can no longer be triggered.

The plan to be fixed $\pi_t = \langle \pi, \langle 0, 2\kappa \rangle \rangle$ is defined in such a way that $|\pi| = |A_\phi| = \kappa$ and for each $i \in \{1, ..., \kappa\}$, the i-th timestamped action has the form $\langle a_{f_i}, 2i-1 \rangle$. It is easy to notice that such a plan is always invalid for Π_ϕ due to the $E_\perp$ events.

Finally, we impose the constraints according to Definition 5, i.e., $\mathcal{C}_\phi = \mathcal{C}_\mathcal{I} \cup \mathcal{C}_\mathcal{S} \cup \mathcal{C}_\mathcal{W}(\omega) \cup \{\langle t'_e \leq 2\kappa + \sigma \rangle\}$ where $\sigma = 0$ and $\omega = 2$.

It is easy to see that the problem thus obtained is polynomial on the formula ϕ.

We have to show that each valid solution to the obtained problem is a solution for SAT, and vice versa. Let $\mathcal{M}$ be a logical model satisfying ϕ, and let us assume that such a model is expressed as a full assignment of the variables $\overline{\mathbb{V}}_\phi$, i.e., $\mathcal{M} = \{\langle f_i := b_i \rangle \mid f_i \in \overline{\mathbb{V}}_\phi\}$ where each $b_i \in \{\perp, \top\}$. $\mathcal{M}$ can be converted into a plan $\pi'_t = \langle \pi_\mathcal{M}, \langle 0, 2 \cdot \kappa \rangle \rangle$ for Π_ϕ. For each $i \in \{1, ..., \kappa\}$, the i-th timestamped action of $\pi_\mathcal{M}$ is defined as $\langle a_i, t_i \rangle$ where $a_i = a_{f_i}$ and $t_i = 2i$ if $b_i = \top$ and $t_i = 2i-2$ otherwise. Each application of a_{f_i} of $\pi_\mathcal{M}$ triggers the event $\varepsilon^+_{f_i}$ ($\varepsilon^-_{f_i}$) that assigns $\top$ ($\perp$) to f_i if such a variable is assigned to $\top$ ($\perp$) in $\mathcal{M}$. It is easy to verify that the truth values of the variables F_ϕ persist once assigned by the events, and no event $E_\perp$ is triggered. It follows that the final state is a full assignment equivalent to the model $\mathcal{M}$ and T remains true. So, such a final state achieves the goal, the actions are always applicable and then the plan is valid for Π_ϕ. It remains to prove that this plan is compliant with constraints $\mathcal{C}$. By construction, the only actions appearing in $\pi_\mathcal{M}$ are those from π, so $\mathcal{C}_\in$ and $\mathcal{C}_\notin$ are satisfied. Again, by construction, $\pi_\mathcal{M}$ preserves the ordering of π, then $\mathcal{C}_\mathcal{S}$ is satisfied. Each assignment of $\mathcal{M}$ is mapped into an action timestamped in $2i-2$ or $2i$, exactly at the extremes of the time windows $\mathcal{TW}(2i-1, 2) = [2i-2, 2i]$, therefore $\mathcal{C}_\mathcal{W}(2)$ is satisfied. Finally, the envelope is compatible with the constraint $\langle t'_e \leq 2\kappa + \sigma \rangle$ since $t'_e = 2\kappa$. For the other direction, let $\pi'_t = \langle \pi', \langle 0, 2\kappa \rangle \rangle$ be a valid solution for the FIXABILITY$^\leq_{\mathcal{WS}}$ problem described so far. Due to the constraints $\mathcal{C}_\mathcal{W}(2)$, in each valid plan π'_t for Π, each action is executed necessarily in $2i$ or $2i-2$, otherwise the goal would be unachievable because of $E_\perp$; so, we can build a logical model for ϕ as $\mathcal{M} = \mathcal{M}_\top \cup \mathcal{M}_\perp$ where:

$$\mathcal{M}_\top = \{\langle f_i := \top \rangle \mid \langle a'_i, t'_i \rangle \in \pi' \wedge t'_i = 2i\}$$
$$\mathcal{M}_\perp = \{\langle f_i := \perp \rangle \mid \langle a'_i, t'_i \rangle \in \pi' \wedge t'_i = 2i-2\}.$$

In a similar way to what is done for the opposite direction of the proof, it is at this point easy to see that this model satisfies ϕ. □

Lemma 3. FIXABILITY$^\leq_\mathcal{I} \in$ NP.

Proof Sketch. Firstly, note that the FIXABILITY$^\leq_\mathcal{I}$ problem $\langle \Pi, \pi_t = \langle \pi, \langle t_s, t_e \rangle \rangle, \mathcal{C} = \mathcal{C}_\mathcal{I} \cup \{\langle t'_e \leq t_e + \sigma \rangle\} \rangle$ can be restated as a decision problem in which the answer is *yes* iff there exists a valid plan π'_t for Π that complies with the constraints $\mathcal{C}$ w.r.t. π_t.

To prove the membership of FIXABILITY$^\leq_\mathcal{I}$ in NP, we design a non-deterministic polynomial algorithm for solving it. This algorithm consists of (i) a *guessing phase* that generates a possible solution for FIXABILITY$^\leq_\mathcal{I}$ under the form of a binary string, and (ii) a *verification phase* that checks whether the guessed solution is a valid one.

(i) Solving FIXABILITY$^\leq_\mathcal{I}$ consists in finding a plan $\pi'_t = \langle \pi', \langle t_s, t'_e \rangle \rangle$ that solves Π and such that all action instances in π appear in π' as well, there are no actions in π' which are not in π, and $t'_e \leq t_e + \sigma$. The problem can be interpreted as a search for a rescheduling of actions π in the interval $[t_s, t_e + \sigma]$ with a granularity of $\delta \in \mathbb{Q}_{>0}$. Note that the number of timestamps at which the action instances can be scheduled is $m = \frac{t_e + \sigma - t_s}{\delta} + 1$ (such points are enumerated in the set $\{t_s + \delta \cdot i \mid i \in \{0, ..., \frac{t_e+\sigma-t_s}{\delta}\}\}$). Therefore, let $\hat{n} = \lceil \log_2(n) \rceil$ and $\hat{m} = \lceil \log_2(m) \rceil$, we can encode a guess for FIXABILITY$^\leq_\mathcal{I}$ as a binary string of the following form:

$$x = x_{actions} \cdot x_{wait} = \overbrace{x^t_{a_1} \cdot x^p_{a_1} \cdot ... \cdot x^t_{a_n} \cdot x^p_{a_n}}^{x_{actions}} \cdot x_{wait},$$

where $x_{actions}$ is a binary representation of π' while x_{wait} of $\langle t_s, t'_e \rangle$. $x_{actions}$ is made up of n sub strings, each of which refers to a_i of π; specifically, $x^t_{a_i}$ and $x^p_{a_i}$ use $\hat{m}$ and $\hat{n}$ bits, respectively. Finally x_{wait} uses $\hat{m}$ bits. Intuitively, $x^t_{a_i}$ encodes the timestamp in which a_i is applied in π' and, since in our temporal model it is possible to execute more than one action in the same timestamp, $x^p_{a_i}$ encodes the position of a_i w.r.t. the timestamp.

From the binary string x we can extract a solution for FIXABILITY$^\leq_\mathcal{I}$ that is $\pi'_t = \langle \pi', \langle t_s, t_s + \delta \cdot (x_{wait})_{10} \rangle \rangle$,[1] where each x_{a_i} of x induces a timestamped action $\langle a_j, t_j \rangle$ in π' where $a_j = a_i$ and $t_j = t_s + \delta \cdot (x^t_{a_i})_{10}$. Furthermore, for each $i \neq j$ such that $x^t_{a_i} = x^t_{a_j}$ and $x^p_{a_i} < x^p_{a_j}$, there are two actions $\langle a_z, t_z \rangle$ and $\langle a_k, t_k \rangle$ in π' where $a_z = a_i$, $a_k = a_i$, $t_k = t_z = t_s + \delta \cdot (x^t_{a_i})_{10}$ and $z < k$ (which means that a_z strictly precedes a_k in π').

Note that the size of a binary string representing a solution for FIXABILITY$^\leq_\mathcal{I}$ is $n \cdot (\lceil log_2(n) \rceil + \lceil log_2(m) \rceil) + \lceil log_2(m) \rceil$ bits.

[1] Given a binary string x, we indicate with $(x)_{10}$ the integer number obtained by representing x in decimal base.

(ii) For the verification phase, firstly, we verify whether x is in a proper format, meaning that it induces a plan for Π. The first condition to be checked is that for each $i, j \in \{1, ..., n\}$ with $i \neq j$, then $x_{a_i} \neq x_{a_j}$; this condition ensures that x induces an absolute ordering among the actions of π'. The second condition to be checked is that $\max(\{x^t_{t_i} \mid i \in \{1, ..., n\}\}) \leq x_{wait}$; this condition ensures that each action of π' is timestamped before $t_s + \delta \cdot (x_{wait})_{10}$. Both of these conditions can be verified in polynomial time and they ensure that x induces a compatible input for FIXABILITY$^{\leq}_{\mathcal{I}}$. If at least one of these tests fails, x is rejected. Otherwise, we verify that the solution π'_t obtained from x is compliant with $\mathcal{C}$ and valid for Π. The verification of the compliance with $\mathcal{C}$ can be performed in polynomial time because the number of constraints is equal to $|\mathcal{C}_{\mathcal{I}}| = |\mathcal{C}_{\in}| + |\mathcal{C}_{\notin}| = |\pi| + |A \setminus \{a \mid \langle a, t\rangle \in \pi\}|$. Finally, we need to evaluate that the plan π'_t is valid for Π. According to Lemma 1, we know that VALIDATION belongs to the complexity class P. Therefore, verifying the validity of π'_t is also a polynomial task. Since both the guessing and verification phases have polynomial complexities, we have successfully developed a polynomial non-deterministic algorithm for solving the FIXABILITY$^{\leq}_{\mathcal{I}}$ problem. □

Theorem 1. *The problem* FIXABILITY$^{\leq}_{Z}$ *with* $Z \in \{\mathcal{I}, \mathcal{S}, \mathcal{W}, \mathcal{WS}\}$ *is* NP-COMPLETE.

Proof Sketch. FIXABILITY$^{\leq}_{\mathcal{WS}}$ NP-Completeness follows by Lemmas 2 and 3. Problems above FIXABILITY$^{\leq}_{\mathcal{WS}}$ can use the same construction used in Lemma 2, as we do not need to have an order among the actions instances, nor windows for which they can be made applicable. Therefore, they are all NP-HARD. Moreover, they also belong to NP, as one can use the same non-deterministic algorithm used for Lemma 3 properly extended. It follows that all problems lying between FIXABILITY$^{\leq}_{\mathcal{WS}}$ and FIXABILITY$^{\leq}_{\mathcal{I}}$ are NP-COMPLETE. □

Solving Plan Fixing Problems via Reformulation

In our work (Percassi, Scala, and Vallati 2022), we demonstrated that solving the PDDL+ planning problem $\Pi^{\pi_t}_{\mathcal{V}_0}$, obtained through the $\mathcal{V}_0$ reformulation, is equivalent to solving VALIDATION. Since VALIDATION can be viewed as a specific case of the general FIXABILITY problem, we address the problems formalised in Definition 5 by suitably relaxing the preconditions of the actions and goals (if necessary) in the planning problem induced by $\mathcal{V}_0$.

In the current section, we will consider the assignment $\Pi = \langle F, X, I, G, A, E, P\rangle$ and $\pi_t = \langle \pi, \langle t_s, t_e\rangle\rangle$ implicit.

The first reformulation that we present is called $\mathcal{R}_{\mathcal{I}}$, and it is designed to solve the FIXABILITY$_{\mathcal{I}}$ problem.

Definition 6 (Reformulation $\mathcal{R}_{\mathcal{I}}$)**.** *Let* $\langle \Pi, \pi_t, \mathcal{C}_{\mathcal{I}}\rangle$ *be a* FIXABILITY$_{\mathcal{I}}$ *problem. The reformulation* $\mathcal{R}_{\mathcal{I}}$ *produces a plan fixing* PDDL+ *problem* $\Pi^{\pi_t}_{\mathcal{R}_{\mathcal{I}}} = \langle F \cup F_A, X, I, G', A_\pi, E, P\rangle$ *such that:*

$$F_A = \{\textit{a-d}_i \mid \langle a_i, t_i\rangle \in \pi\}$$

$$A_\pi = \bigcup_{\langle a_i, t_i\rangle \in \pi} \{\langle \textit{pre}(a_i) \wedge \neg \textit{a-d}_i, \textit{eff}(a_i) \cup \{\textit{a-d}_i\}\rangle\}$$

$$G' = G \wedge \bigwedge_{\langle a_i, t_i\rangle \in \pi} \textit{a-d}_i$$

$\mathcal{R}_{\mathcal{I}}$ inherits the same structure of $\mathcal{V}_0$ but relaxes the timestamps and execution ordering constraints over the actions A_π. The actions in A_π are relaxed in the sense that they do not require to be executed in the timestamps prescribed by π_t. Furthermore, the additional Boolean variables F_A are not used to impose an absolute ordering among A_π, but just to ensure that each action can be executed at most once. G' requires that G be satisfied and that all actions be performed.

Now we move on to a reformulation for answering the FIXABILITY$_{\mathcal{W}}$ problem, but before doing so we need some additional notation. Given the timestamp t_i associated with the i-th action in π, we indicate with $prev(t_i, \omega) = \{j \mid \langle a_j, t_j\rangle \in \pi \wedge t_j + \omega < t_i\}$ the set of the indexes of all the actions temporally preceding a_i of at least a quantity equal to ω.

Definition 7 (Reformulation $\mathcal{R}_{\mathcal{W}}$)**.** *Let* $\omega \in \mathbb{Q}_{\geq 0}$ *and let* $\langle \Pi, \pi_t, \mathcal{C}_{\mathcal{I}} \cup \mathcal{C}_{\mathcal{W}}(\omega)\rangle$ *be a* FIXABILITY$_{\mathcal{W}}$ *problem. The reformulation* $\mathcal{R}_{\mathcal{W}}$ *produces a plan fixing* PDDL+ *problem* $\Pi^{\pi_t}_{\mathcal{R}_{\mathcal{W}}} = \langle F \cup F_A \cup \{\top\}, X \cup \{\textit{time}\}, I \cup \{\langle \textit{time} := t_s\rangle\}, G', A_\pi, E, P \cup \{\rho_{\textit{time}}\}\rangle$ *having the same structure of* $\Pi^{\pi_t}_{\mathcal{R}_{\mathcal{I}}}$ *except for the set of actions* A_π *that is reshaped as follows:*

$$A_\pi = \bigcup_{\langle a_i, t_i\rangle \in \pi} \{\langle \textit{pre}(a_i) \wedge \neg \textit{a-d}_i \wedge \bigwedge_{j \in prev(t_i, \omega)} \textit{a-d}_j \wedge \langle \textit{time} \geq \mathcal{TW}_l(t_i, \omega)\rangle \wedge \langle \textit{time} \leq \mathcal{TW}_u(t_i, \omega)\rangle, \textit{eff}(a_i) \cup \{\textit{a-d}_i\}\rangle\}$$

Note that $\mathcal{R}_{\mathcal{W}}$ needs to use a process that is always active and keeps track of the elapsed time in the numeric variable *time*, i.e., $\rho_{\textit{time}} = \langle \top, \{\langle \textit{time}, 1\rangle\}\rangle$. Similarly to $\mathcal{R}_{\mathcal{I}}$, $\mathcal{R}_{\mathcal{W}}$ relaxes the constraints about the ordering of the actions but it additionally extends the precondition of each action in A_π with a formula involving the variable *time*. This formula imposes that each action $a'_i \in A_\pi$, originally executed at t_i in π, must be executed in π' within the time window $\mathcal{TW}(t_i, \omega)$.

Furthermore, the preconditions of each $a'_i \in A_\pi$ are extended with the formula $\bigwedge_{j \in prev(t_i)} \textit{a-d}_j$. Such a condition additionally requires that before executing a'_i within $\mathcal{TW}(t_i, \omega)$, it is necessary to execute all actions a_j from π with $j \in prev(t_i, \omega)$; these are the actions that temporally precede a_i by at least ω. Note that these constraints are redundant since those that require *time* $\in [\mathcal{TW}_l(t_i, \omega), \mathcal{TW}_u(t_i, \omega)]$ already ensure this property. Their only purpose is to speed up the planning process.

Now we move on to a reformulation for addressing the FIXABILITY$_{\mathcal{S}}$ problem. This reformulation enforces the ordering given by π whilst leaving to the agent the freedom to execute actions in any timestamp (as $\mathcal{R}_{\mathcal{I}}$ does).

Definition 8 (Reformulation $\mathcal{R}_\mathcal{S}$)**.** *Let $\langle \Pi, \pi_t, \mathcal{C}_\mathcal{I} \cup \mathcal{C}_\mathcal{S} \rangle$ be a* FIXABILITY$_\mathcal{S}$ *problem. The reformulation $\mathcal{R}_\mathcal{S}$ produces a plan fixing* PDDL+ *problem $\Pi^{\pi_t}_{\mathcal{R}_\mathcal{S}} = \langle F \cup F_A, X, I \cup \{a\text{-}d_0\}, G \wedge a\text{-}d_n, A_\pi, E, P \rangle$ such that:*

$$F_A = \{a\text{-}d_i \mid \langle a_i, t_i \rangle \in \pi\}$$

$$A_\pi = \bigcup_{\langle a_i, t_i \rangle \in \pi} \{\langle pre(a_i) \wedge a\text{-}d_{i-1} \wedge \neg a\text{-}d_i, eff(a_i) \cup \{a\text{-}d_i\} \rangle\}$$

Differently from $\mathcal{R}_\mathcal{I}$, $\mathcal{R}_\mathcal{S}$ relaxes only timestamp constraints of $\mathcal{V}_0$, and leaves the ordering constraints untouched.

The constraints introduced by $\mathcal{R}_\mathcal{W}$ and $\mathcal{R}_\mathcal{S}$ can be combined in order to obtain a further constrained problem. We denote such a composition $\mathcal{R}_{\mathcal{WS}} = \mathcal{R}_\mathcal{W} \circ \mathcal{R}_\mathcal{S}$. The resulting planning problem $\Pi^{\pi_t}_{\mathcal{R}_{\mathcal{WS}}}$ can be used for solving the FIXABILITY$_{\mathcal{WS}}$ problem.

We now show how the previous reformulations can be adapted to handle temporally bounded FIXABILITY problems.

Definition 9 (Temporal Bounded Reformulations $\mathcal{R}^{\leq}_Z$)**.** *Let $Z \in \{\mathcal{I}, \mathcal{S}, \mathcal{W}, \mathcal{WS}\}$ and let $\langle \Pi, \pi_t, \mathcal{C}_Z \rangle$ be a* FIXABILITY$_Z$ *problem. Let $\Pi^{\pi_t}_{\mathcal{R}_Z} = \langle F', X', I', G', A', E', P' \rangle$ be the plan fixing* PDDL+ *problem obtained by $\mathcal{R}_Z$. Let $\sigma \in \mathbb{Q}_{\geq 0}$ and let $\langle \Pi, \pi_t, \mathcal{C}_Z \cup \{\langle t'_e \leq t_e + \sigma \rangle\} \rangle$ be the temporally bounded problem* FIXABILITY$^{\leq}_Z$ *obtained from $\langle \Pi, \pi_t, \mathcal{C}_Z \rangle$; the reformulation $\mathcal{R}^{\leq}_Z$ produces a plan fixing* PDDL+ *problem $\Pi^{\pi_t}_{\mathcal{R}^{\leq}_Z} = \langle F' \cup \{\mathrm{T}\}, X' \cup \{time\}, I' \cup \{\langle time := t_s \rangle\}, G' \wedge \langle time \leq t_e + \sigma \rangle, A', E', P' \cup \{\rho_{time}\} \rangle$.*

In other words, $\mathcal{R}^{\leq}_Z$ inherits the structure of $\mathcal{R}_Z$ and extends the goal formula by requiring that the elapsed time is at most $t_e + \sigma$ ($\langle time \leq t_e + \sigma \rangle$). This reformulation, regardless of the chosen Z, imposes a bound on the elapsed time, preventing the exploration of states having time exceeding a certain threshold controlled by σ.

Properties

This section discusses the soundness and the completeness of the proposed reformulations w.r.t. the FIXABILITY problems they are associated with. We focus on a particular case, i.e., $\mathcal{R}_\mathcal{W}$; the other proofs can be easily derived from this one.

Theorem 2 (Soundness and Completeness)**.** *Let Π be a* PDDL+ *problem, let π_t a plan for Π and let $Z \in \{\mathcal{I}, \mathcal{S}, \mathcal{W}, \mathcal{WS}\}$.* FIXABILITY$_Z$ *admits a solution iff $\Pi^{\pi_t}_{\mathcal{R}_Z}$ does so.*

Proof Sketch. ($\Rightarrow$) To prove the completeness of $\mathcal{R}_\mathcal{W}$, we observe that the existence of a solution of a FIXABILITY$_\mathcal{W}$ problem $\langle \Pi, \pi_t = \langle \pi, \langle t_s, t_e \rangle \rangle, \mathcal{C} = \mathcal{C}_\mathcal{I} \cup \mathcal{C}_\mathcal{W}(\omega) \rangle$ implies that there exists a valid plan for $\Pi^{\pi_t}_{\mathcal{R}_\mathcal{W}}$. We show this by construction.

Let $\pi'_t = \langle \pi', \langle t_s, t'_e \rangle \rangle$ be a solution for FIXABILITY$_\mathcal{W}$, we construct a plan $\pi''_t = \langle \pi'', \langle t_s, t'_e \rangle \rangle$ for $\Pi^{\pi_t}_{\mathcal{R}_\mathcal{W}}$ as follows: for each i-th timestamped action $\langle a'_i, t'_i \rangle$ in π', there is a corresponding $\langle a''_i, t'_i \rangle$ in π' where the action $a''_i \in A_\pi$ is the action $a'_i \in A$ extended with the preconditions and effects introduced by $\mathcal{R}_\mathcal{W}$. Let $\mathbb{H}$ and $\mathbb{H}'$ be the PDDL+ plan projections of π_t and π'_t, and let τ and τ' be the sequences of states associated to each STP of the two histories. Since $\mathcal{R}_\mathcal{W}$ reformulates Π without changing the goals, the processes and the events, we get that τ and τ' are equivalent if we consider the restricted set of variables $F \cup X$. Now, what we need to prove is that the actions of π'' are applicable in τ'. If π'_t is a solution of FIXABILITY$_\mathcal{W}$ then each action $\langle a'_i, t'_i \rangle$ in π' satisfies the constraint $time(t_i, \omega, \pi') \in \mathcal{C}_\mathcal{W}(\omega)$. This constraints prescribes that $t'_i \in \mathcal{TW}(t_i, \omega)$. Such an action is transformed, according to the mapping described above, into the action $\langle a''_i, t'_i \rangle$ in π'', which requires to be applied when *time* $\in \mathcal{TW}(t_i, \omega)$. Since ρ_{time} synchronises the *time* variable with the actual elapsed time, we get that for a given state s in which a''_i is applied, then $s \models$ *time* $\in \mathcal{TW}(t_i, \omega)$ and then a''_i is applicable in s. Finally, since $\mathbb{H}'$ achieves G' and each action is applicable, π''_t is valid for $\Pi^{\pi_t}_{\mathcal{R}_\mathcal{W}}$.

($\Rightarrow$) Similarly to the other direction, to prove that $\mathcal{R}_\mathcal{W}$ is sound, we show that each valid plan for $\Pi^{\pi_t}_{\mathcal{R}_\mathcal{W}}$ implies that there is a valid plan for FIXABILITY$_\mathcal{W}$. And we do so again by construction. Let $\pi'_t = \langle \pi', \langle t_s, t'_e \rangle \rangle$ be a valid plan for $\Pi^{\pi_t}_{\mathcal{R}_\mathcal{W}}$, we construct a plan $\pi''_t = \langle \pi'', \langle t_s, t'_e \rangle \rangle$ for Π as follows: for each i-th timestamped action $\langle a'_i, t'_i \rangle$ in π' there is a corresponding $\langle a''_i, t'_i \rangle$ in π'' where $a''_i \in A$ is the action $a'_i \in A_\pi$ deprived of the additional preconditions and effects added by $\mathcal{R}_\mathcal{W}$. To prove that the plan thus obtained is a solution for FIXABILITY$_\mathcal{W}$, we have to show that (i) the plan is valid for Π and that (ii) it satisfies the constraints $\mathcal{C} = \mathcal{C}_\mathcal{I} \cup \mathcal{C}_\mathcal{W}(\omega)$. (i) Since Π is a reduced version of $\Pi^{\pi_t}_{\mathcal{R}_\mathcal{W}}$, it is easy to notice that the operators of π'' are still applicable if their correspondents of $\mathcal{R}_\mathcal{W}$ are, and numeric variables change as time passes as they do in $\mathcal{R}_\mathcal{W}$. Then, π''_t is valid for Π. (ii) Note that $G' \models \bigwedge_{\langle a_i, t_i \rangle \in \pi} a\text{-}d_i$. So, since π'_t is valid for $\Pi^{\pi_t}_{\mathcal{R}_\mathcal{W}}$, π' contains each action from A_π. Since π' is mapped into π'' and A_π is generated by using exclusively the action from π, we get that π''_t complies with the constraints $\mathcal{C}_\mathcal{I}$. Each action $\langle a'_i, t'_i \rangle$ of π' is applicable iff *time* $\in \mathcal{TW}(t_i, \omega)$ and since ρ_{time} synchronises *time* with the actual elapsed time we know that $t'_i \in \mathcal{TW}(t_i, \omega)$. Each action from π' maps into an action $\langle a''_i, t'_i \rangle$ of π'', so it easy to see that π''_t complies with $\mathcal{C}_\mathcal{W}(\omega)$. □

FIX $\langle \Pi, \pi_t, \mathcal{C} \rangle$	Constraints $\mathcal{C}$	Complexity	FIX as PDDL+ Planning
FIX$_\emptyset$	$\emptyset$	Undecidable	Π
FIX$_\mathcal{I}$	$\mathcal{C}_\mathcal{I}$	NP-HARD	$\Pi^{\pi_t}_{\mathcal{R}_\mathcal{I}}$
FIX$_\mathcal{S}$	$\mathcal{C}_\mathcal{I} \cup \mathcal{C}_\mathcal{S}$		$\Pi^{\pi_t}_{\mathcal{R}_\mathcal{S}}$
FIX$_\mathcal{W}$	$\mathcal{C}_\mathcal{I} \cup \mathcal{C}_\mathcal{W}(\omega)$		$\Pi^{\pi_t}_{\mathcal{R}_\mathcal{W}}$
FIX$_{\mathcal{WS}}$	$\mathcal{C}_\mathcal{I} \cup \mathcal{C}_\mathcal{W}(\omega) \cup \mathcal{C}_\mathcal{S}$		$\Pi^{\pi_t}_{\mathcal{R}_{\mathcal{WS}}}$
FIX$^{\leq}_Z$	$\mathcal{C}_Z \cup \{\langle t'_e \leq t_e + \sigma \rangle\}$	NP-COMPLETE	$\Pi^{\pi_t}_{\mathcal{R}^{\leq}_Z}$
VAL	$\mathcal{C}_\mathcal{I} \cup \mathcal{C}_\mathcal{W}(0) \cup$ $\mathcal{C}_\mathcal{S} \cup \{\langle t'_e = t_e \rangle\}$	P	$\Pi^{\pi_t}_{\mathcal{V}_0}$

Table 1: Theoretical results about FIXABILITY (shortened as FIX) and VALIDATION (shortened as VAL) problems. For temporally bounded FIXABILITY problems, $Z \in \{\mathcal{I}, \mathcal{S}, \mathcal{W}, \mathcal{WS}\}$.

A*(h^{max})					
Domain	P	$\mathcal{R}_\mathcal{I}$	$\mathcal{R}_\mathcal{S}$	$\mathcal{R}_\mathcal{W}$	$\mathcal{R}_{\mathcal{WS}}$
BAXTER	**17**	16	**17**	**17**	**17**
DESCENT	8	**20**	**20**	**20**	**20**
HVAC	**16**	2	**16**	**16**	**16**
INTERCEPT	6	8	**10**	9	**10**
LIN-CAR	21	2	**24**	**24**	**24**
LIN-GEN	4	4	**10**	4	6
NON-LIN-CAR	7	6	15	**20**	**20**
OT-CAR	5	5	8	11	**12**
ROVER	**20**	**20**	**20**	**20**	**20**
UTC	0	0	7	3	**10**
Σ	104	83	147	144	**155**
A*(h^{add})					
Domain	P	$\mathcal{R}_\mathcal{I}$	$\mathcal{R}_\mathcal{S}$	$\mathcal{R}_\mathcal{W}$	$\mathcal{R}_{\mathcal{WS}}$
BAXTER	**19**	17	17	17	17
DESCENT	9	**20**	**20**	**20**	**20**
HVAC	**16**	5	**16**	**16**	**16**
INTERCEPT	6	8	**10**	**10**	**10**
LIN-CAR	22	2	**24**	**24**	**24**
LIN-GEN	4	6	**10**	4	6
NON-LIN-CAR	9	6	14	**20**	**20**
OT-CAR	5	5	11	15	**16**
ROVER	**20**	**20**	**20**	**20**	**20**
UTC	0	1	19	15	**20**
Σ	110	90	161	161	**169**
UPMURPHI					
Domain	P	$\mathcal{R}_\mathcal{I}$	$\mathcal{R}_\mathcal{S}$	$\mathcal{R}_\mathcal{W}$	$\mathcal{R}_{\mathcal{WS}}$
BAXTER	12	15	**17**	16	**17**
DESCENT	10	**17**	**17**	**17**	**17**
HVAC	8	3	**16**	5	**16**
LIN-CAR	2	0	7	3	**22**
LIN-GEN	1	2	**6**	2	**6**
ROVER	5	5	**20**	**20**	**20**
Σ	38	42	83	63	**98**

Table 2: Coverage comparison between planning from scratch (P) and plan fixing with different planning engines.

Experimental Analysis

The aim of our analysis is to assess the usefulness of different plan fixing reformulations over replanning from scratch in the context of discretised PDDL+. Firstly, we consider synthetically perturbed plans; then, we focus on the setting of generating plans using increasingly small discretisation steps, to provide more accurate solutions over time; this is to simulate a plan, execution, and replan framework where plans are not valid in execution and need to be refined.

As benchmarks, we use domains from Percassi, Scala, and Vallati (2022) extended with two additional ones: INTERCEPT and NON-LIN-CAR, both taken from Scala et al. (2016). For LIN-CAR we used larger instances to emphasise the computational aspect (Kuroiwa, Shleyfman, and Beck 2022). For each domain, we consider between 10 and 25 instances, with a corresponding initial solution plan generated with $\delta = 1$, unless differently specified. The translator and the benchmark suite can be found at https://bit.ly/30gMyNW. All experiments run on an Intel Xeon Gold 6140M CPUs with 2.30 GHz, with a cutoff time of $1,800$ seconds, and 8GB of RAM.

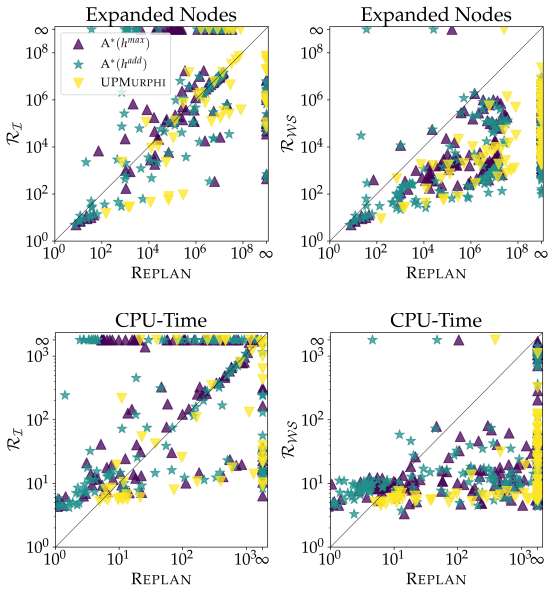

Figure 2: Expanded Nodes and CPU-Time of replanning from scratch (REPLAN) versus plan fixing for $\mathcal{R}_\mathcal{I}$ and $\mathcal{R}_{\mathcal{WS}}$.

Perturbed Plans. To evaluate the computational aspects of the introduced reformulations for plan fixing, we took all instances of our benchmark suite, solve them, and invalidate the resulting plans. The invalidity is injected by perturbing actions' timestamps within a given μ, which depends on the original plan and on the characteristics of the domain model.

Given a valid plan π_t, we extract the vector of timestamps at which actions are performed, along with the envelope of the plan. This results in a vector $\langle t_1, ..., t_m \rangle$. We then calculate the distance vector $D = \langle t_2 - t_1, ..., t_m - t_{m-1} \rangle$ and compute the average distance $\mu = \text{average}(D)$. Next, we perturb each timestamp t_i of the plan π_t using uniformly distributed noise in the range $[t_i - \frac{\mu}{2}, t_i + \frac{\mu}{2}]$. Since we want to measure the computational load of fixing plans with different types of constraints, we (i) ensure that the action ordering of the original plan is preserved w.r.t. the injected noise, and (ii) define the plan fixing problems by imposing a time window of $\omega = \mu$ when it is necessary. This ensures that the plans are always fixable in this experiment.

For a good overview across different PDDL+ approaches, we consider 3 planning engines: UPMURPHI (Penna, Magazzeni, and Mercorio 2012), and A* with the h^{add} and h^{max} heuristics provided by ENHSP (https://sites.google.com/view/enhsp/). All experiments are run with $\delta = 1$ but for INTERCEPT, where a $\delta = 0.1$ is needed.

Table 2 shows the coverage achieved by the different planning systems in both planning from scratch (P) and fixing an initial plan using the introduced reformulations. We have omitted domains where a planning system failed to solve any instances. The results indicate that plan fixing can be highly advantageous across all considered planning ap-

	BAXTER				
	Coverage by δ			avg(CPU-Time) by δ	
	1	0.1	0.01	0.1	0.01
ONE-SHOT	19	12	11	4.4	6.2
FIX-PLAN($\mathcal{R}_\mathcal{I}$)	19	19	19	14.3	25.3
FIX-PLAN($\mathcal{R}_\mathcal{S}$)	19	19	19	14.7	25.8
	DESCENT				
ONE-SHOT	9	0	0	—	—
FIX-PLAN($\mathcal{R}_\mathcal{I}$)	9	4	1	293.3	214.7
FIX-PLAN($\mathcal{R}_\mathcal{S}$)	9	4	1	259.9	172.2
	INTERCEPT				
ONE-SHOT	—	7	0	332.1	—
FIX-PLAN($\mathcal{R}_\mathcal{I}$)	—	7	7	332.1	379.2
FIX-PLAN($\mathcal{R}_\mathcal{S}$)	—	7	7	332.1	377.7
	NON-LIN-CAR				
ONE-SHOT	8	0	0	—	—
FIX-PLAN($\mathcal{R}_\mathcal{I}$)	8	1	0	416.8	—
FIX-PLAN($\mathcal{R}_\mathcal{S}$)	8	1	0	108.9	—
	OT-CAR				
ONE-SHOT	5	0	0	—	—
FIX-PLAN($\mathcal{R}_\mathcal{I}$)	5	0	0	—	—
FIX-PLAN($\mathcal{R}_\mathcal{S}$)	5	1	0	1343.2	—
	ROVER				
ONE-SHOT	20	0	0	—	—
FIX-PLAN($\mathcal{R}_\mathcal{I}$)	20	20	20	438.9	456.8
FIX-PLAN($\mathcal{R}_\mathcal{S}$)	20	20	20	451.8	468.5

Table 3: Coverage and average CPU-Time of A* with h^{add} in planning from scratch (ONE-SHOT) and in plan fixing (FIX-PLAN) using either $\mathcal{R}_\mathcal{I}$ or $\mathcal{R}_\mathcal{S}$ to accommodate the finer δ, i.e., 0.1 and 0.01; for plan fixing, the reported average CPU-Time is the accumulated time.

proaches, particularly when using $\mathcal{R}_\mathcal{S}$, $\mathcal{R}_\mathcal{W}$, and $\mathcal{R}_{\mathcal{WS}}$. This is not surprising, considering that the different reformulations encode plan fixing problems with an increasing number of constraints and that they can rely on the knowledge provided under the form of the plan to be fixed. Surprisingly, $\mathcal{R}_\mathcal{I}$ is preferable to planning from scratch when we use UP-MURPHI, indicating that these characterisations could even be more beneficial over non-heuristic planning engines.

To shed some light on the behaviour of the systems when the most ($\mathcal{R}_{\mathcal{WS}}$) and the least ($\mathcal{R}_\mathcal{I}$) constrained reformulations are used, Figure 2 shows how CPU-Time and number of expansions vary in comparison to planning from scratch. Our pairwise comparison indeed confirms that $\mathcal{R}_{\mathcal{WS}}$ largely outperforms replanning from scratch, while the speed up provided by $\mathcal{R}_\mathcal{I}$ is much more limited.

Incrementally Finer-grained Discretisation. Here we are interested in assessing if plan fixing reformulations can be used effectively to produce incrementally valid plans for smaller discretisation steps in an anytime fashion, hence mimicking a plan, execution, replan scenario where more accurate plans are needed over time. Starting from a plan generated with $\delta = 1$, we try to fix it using the less constrained reformulations ($\mathcal{R}_\mathcal{I}$ or $\mathcal{R}_\mathcal{S}$) to generate plans that are valid for smaller discretisation steps in a sequence of planning episodes. In our evaluation, we consider the sequence $\delta_{seq} = \langle 1, 0.1, 0.01 \rangle$ and we compare the CPU-Time needed for computing a solution *directly* for each $\delta \in \delta_{seq}$ (ONE-SHOT), and the accumulated time needed by a system, namely FIX-PLAN, that first computes a solution from scratch with the highest δ, and then try to fix such a plan to work also with the finer δ, i.e., 0.1 and 0.01. Therefore, ONE-SHOT performs a single search episode while FIX-PLAN performs at most three episodes. Both approaches use A$^*(h^{add})$ as a search scheme; other configurations show a similar trend.

Table 3 shows the results of the analysis over the domains where at least one solution for δ smaller than 1 is found by any of the approaches. Results are presented in terms of coverage and average CPU-Time over the solved instances. In the case of plan fixing approaches, the CPU-Time is accumulated over the subsequent calls to solve an instance with finer discretisation steps. In most domains, the plan-fixing-based approach can help generate plans that are valid for finer δ values, hence increasing the accuracy of the simulated dynamics.

Conclusions

To support the use of PDDL+ planning in complex real-world scenarios, we have introduced the concept of discretised PDDL+ plan fixing. Our work has demonstrated that plan fixing can address a wide range of problems, from planning to validation, and we have introduced a set of reformulations that enable each such problem to be solved using a planner-independent approach. Our empirical analysis indicates that plan fixing is a viable alternative to planning from scratch, and can help generate plans for fine-grained discretisations.

Future work will focus on extending plan fixing to additional circumstances and leveraging plan fixing to support efficient plan repair in the presence of dedicated optimisation functions. Another possible avenue for future work is to study whether there are upper bounds for the complexity and decidability of the unbounded fragments entailed by our formulations, such as FIXABILITY$_\mathcal{I}$. Finally, we plan to extend this approach to allow for the removal and addition of actions to the plan to be fixed.

Acknowledgements

Francesco Percassi and Mauro Vallati were supported by a UKRI Future Leaders Fellowship [grant number MR/T041196/1]. Enrico Scala has been partially supported by AIPlan4EU, a project funded by EU Horizon 2020 research and innovation programme under GA n. 101016442.

References

Arangú, M.; Garrido, A.; and Onaindia, E. 2008. A General Technique for Plan Repair. In *Proceedings of the Twentieth International Conference on Tools with Artificial Intelligence, ICTAI 2008*, 515–518.

Brenner, M.; and Nebel, B. 2009. Continual planning and acting in dynamic multiagent environments. *Autonomous Agents and Multi-Agent Systems*, 19(3): 297–331.

desJardins, M.; Durfee, E. H.; Jr., C. L. O.; and Wolverton, M. 1999. A Survey of Research in Distributed, Continual Planning. *AI Magazine*, 20(4): 13–22.

Fox, M.; Gerevini, A.; Long, D.; and Serina, I. 2006. Plan Stability: Replanning versus Plan Repair. In *Proceedings of the Sixteenth International Conference on Automated Planning and Scheduling, ICAPS 2006*, 212–221.

Fox, M.; Howey, R.; and Long, D. 2005. Validating Plans in the Context of Processes and Exogenous Events. In *Proceedings of the Twentieth National Conference on Artificial Intelligence, AAAI 2005*, 1151–1156.

Fox, M.; and Long, D. 2006. Modelling Mixed Discrete-Continuous Domains for Planning. *Journal of Artificial Intelligence Research*, 27: 235–297.

Gerevini, A.; and Serina, I. 2010. Efficient Plan Adaptation through Replanning Windows and Heuristic Goals. *Fundamenta Informaticae*, 102(3-4): 287–323.

Kuroiwa, R.; Shleyfman, A.; and Beck, J. C. 2022. LM-Cut Heuristics for Optimal Linear Numeric Planning. In *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling, ICAPS 2022*, 203–212.

McCluskey, T. L.; and Porteous, J. M. 1997. Engineering and Compiling Planning Domain Models to Promote Validity and Efficiency. *Artificial Intelligence*, 95(1): 1–65.

Nebel, B.; and Koehler, J. 1995. Plan Reuse Versus Plan Generation: A Theoretical and Empirical Analysis. *Artificial Intelligence*, 76(1-2): 427–454.

Penna, G. D.; Magazzeni, D.; and Mercorio, F. 2012. A universal planning system for hybrid domains. *Applied Intelligence*, 36(4): 932–959.

Percassi, F.; Scala, E.; and Vallati, M. 2022. The Power of Reformulation: From Validation to Planning in PDDL+. In *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling, ICAPS 2022*, 288–296.

Percassi, F.; Scala, E.; and Vallati, M. 2023. A Practical Approach to Discretised PDDL+ Problems by Translation to Numeric Planning. *Journal of Artificial Intelligence Research*, 76: 115–162.

Piotrowski, W. M.; Fox, M.; Long, D.; Magazzeni, D.; and Mercorio, F. 2016. Heuristic Planning for Hybrid Systems. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI 2016*, 4254–4255.

Scala, E. 2014. Plan Repair for Resource Constrained Tasks via Numeric Macro Actions. In *Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling, ICAPS 2014*, 280–288.

Scala, E.; Haslum, P.; Thiébaux, S.; and Ramírez, M. 2016. Interval-Based Relaxation for General Numeric Planning. In *Proceedings of the Twenty-Second European Conference on Artificial Intelligence, ECAI 2016*, volume 285, 655–663.

Scala, E.; and Torasso, P. 2015. Deordering and Numeric Macro Actions for Plan Repair. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*, 1673–1681.

Shin, J.; and Davis, E. 2005. Processes and continuous change in a SAT-based planner. *Artificial Intelligence*, 166(1-2): 194–253.

van der Krogt, R.; and de Weerdt, M. 2005. Plan Repair as an Extension of Planning. In *Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling, ICAPS 2005*, 161–170.