On Translation-Based Approaches from Discrete PDDL+ to Numeric Planning

Francesco Percassi,[1] **Enrico Scala,**[2] **Mauro Vallati,**[1]
[1] School of Computing and Engineering, University of Huddersfield, UK
[2] Dipartimento di Ingegneria dell'Informazione, Università degli Studi di Brescia, Italy
f.percassi@hud.ac.uk, enrico.scala@unibs.it, m.vallati@hud.ac.uk

Abstract

PDDL+ is an expressive planning formalism that enables the modelling of hybrid domains having both discrete and continuous dynamics, in which the agent and the environment description are sharply separated. These models are notoriously difficult to address, and many planning systems do so by assuming discrete semantics. Following a long trend aimed at reformulating complex problems into ones expressed in simpler and more manageable formalisms, translations from discrete PDDL+ into numeric PDDL2.1 have recently been proposed. In this work, firstly we study the existing translations by comparing them from the point of view of induced searches and the size of generated numeric tasks. Secondly, building on the existing translations, we propose a novel translation whose design goes in the direction of defining a scheme sensitive to the structure of the problem.

Introduction

Automated planning is a solid branch of artificial intelligence that aims at designing methodologies for synthesising a sequence of actions capable to transform a given state, i.e., the initial state, into the desired state, i.e., the goal state.

Many real-world systems are *hybrid* in nature, as they are characterised by the coexistence of discrete and continuous state variables, together with changes that can happen over a continuous timeline. In automated planning, hybrid systems can be represented and modelled using the PDDL+ formalism (Fox and Long 2006). PDDL+ provides a representation that puts together an agent, via an action-oriented formalisation, with an explicit representation of the environment and its exogenous dynamics given in terms of processes and events. However, PDDL+ planning problems are notoriously difficult to be solved, and there is a restricted set of planning engines that can natively support them.

Recently, to increase the pool of planning engines that can tackle PDDL+ planning problems, an exponential (EXP) and a polynomial (POLY) translation have been proposed (Percassi, Scala, and Vallati 2021c). Such translations aim to transform a time discretisation of a PDDL+ problem into a simpler formalism, more precisely, a numeric planning problem represented using the PDDL2.1 language (Fox and Long 2003). Both schemata compile the environment into the agent's actions. EXP does so anticipating all possible processes with an exponential encoding that keeps the height of the search tree limited. POLY avoids such an exponential behaviour by unrolling all processes into several consecutive actions but makes the search tree much deeper. Both translations belong to a broader family of approaches aimed at solving problems represented in some language using solvers and tools devoted to tackling problems expressed in some other, less complicated, language (e.g., (Keyder and Geffner 2009; Palacios and Geffner 2009; Cooper, Maris, and Régnier 2010; Taig and Brafman 2013; Grastien and Scala 2017; Percassi and Gerevini 2019; Bonassi et al. 2021; Scala and Grastien 2021)).

One of the challenges of these approaches is to be able to provide compact and efficient translations, and exactly with this objective in mind, a third translation, namely POLY$^-$, has been recently proposed (Percassi, Scala, and Vallati 2021a). Intuitively, POLY$^-$ leverages the advantages of EXP while avoiding the exponential blow up with a schema that ignores some of the possible transitions. This results in a schema that, differently from EXP and POLY which are both sound and complete, is sound but complete only for a subclass of PDDL+.

In order to improve our understanding of these translations, in this paper, we report on a study of the existing translations focusing on the induced state space and the size of the numeric translated tasks, aspects of crucial importance from a practical point of view. Then we propose a new translation, namely EXP$^{\mathcal{L}}$ that, by exploiting the structure of the problem at hand, combinatorially combines only those processes deemed *necessary*, resulting in a schema that is exponential only the size of the largest set of processes affecting a numeric variable. This novel translation is compared from a theoretical point of view with the existing ones, showing when it is generally preferable. These considerations are finally supported by an experimental analysis of optimal planning problems.

Background

In this section we report on the PDDL+ problem (Fox and Long 2006) interpreted over a discrete timeline (Percassi, Scala, and Vallati 2021c), and the problem of numeric planning as the one that can be specified in PDDL2.1 (level 2) (Fox and Long 2003). We borrow notation and the semantics from (Percassi, Scala, and Vallati 2021c), and focus here on the description of those aspects that are crucial to understand this work.

We specify our problems using propositional formulas over numeric and Boolean conditions defined over sets of numeric and Boolean variables. A numeric condition is of the form $\langle \xi \bowtie 0 \rangle$ with ξ being a numeric expression, and $\bowtie \in \{\leq, <, =, >, \geq\}$. A Boolean condition is of the form $f = \{\top, \bot\}$ with f being a Boolean variable.

Definition 1 (PDDL+ problem). *A* PDDL+ *planning problem* Π *is the tuple* $\langle F, X, I, G, A, E, P \rangle$ *in which each element is detailed in the following.* F *and* X *are the Boolean and numeric variables. Numeric variables take values from* $\mathbb{R}$. 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 conditional effects of the form* $c \triangleright e$. *Each conditional effect* $c \triangleright e$ *is such that (i)* c *is a formula and (ii)* e *is a set of Boolean assignments of the form* $\langle f := \{\bot, \top\} \rangle$ *or numeric assignments of the form* $\langle \{asgn, inc, dec\}, x, \xi \rangle$ *where* ξ *is a numeric expression.* P *is a set of processes. A process is a pair* $\langle p, e' \rangle$ *where* p *is a formula and* e' *is a set of numeric continuous effects expressed as pairs* $\langle x, \xi \rangle$ *where* ξ *is the net derivative of* x.

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 and assignments of the form $\langle f = \bot \rangle$ ($\langle f := \bot \rangle$) and $\langle f = \top \rangle$ ($\langle f := \top \rangle$) are shortened to f and $\neg f$, and conditional effects of the form $\top \triangleright e$ are rewritten as e.

A plan for a PDDL+ problem is an ordered set of timed actions plus a time envelope, organised formally as follows.

Definition 2 (PDDL+ plan). *A* PDDL+ plan π_t *is a pair* $\langle \pi, \langle t_s, t_e \rangle \rangle$ *where:* $\pi = \langle \langle a_0, t_0 \rangle, ..., \langle a_{n-1}, t_{n-1} \rangle \rangle$, *with* $t_i \in \mathbb{Q}$, *is a sequence of time-stamped actions;* $\langle t_s, t_e \rangle$, *with* $t_s, t_e \in \mathbb{Q}$ *and* $t_s \leq t_e$, *is the envelope within which* π *is performed.*

We say that π_t is well-formed iff $\forall\, i, j \in [0..n-1]$ and $i < j$, then $t_i \leq t_j$ and $t_s \leq t_i \leq t_e$ hold. Hereinafter we consider just well-formed plans.

PDDL2.1 is the subclass of PDDL+ where we only have actions and no explicit management of time. This is syntactically reflected in the absence of events and processes.

Definition 3 (PDDL2.1 problem). *A* PDDL2.1 problem Π *is a tuple* $\langle F, X, I, G, A, c \rangle$ *where all elements are as for* PDDL+, *yet there are neither processes nor events and* c *associates to each action a rational cost.*

Definition 4 (PDDL2.1 plan). *A* PDDL2.1 plan *is a sequence of actions* $\langle a_0, ..., a_{n-1} \rangle$. *The cost of the plan* π *is the sum of all action costs in* π, *cost*$(\pi) = \sum_{a \, in \, \pi} c(a)$.

Intuitively, a PDDL+ problem consists in finding plans along a potentially infinite timeline, whilst conforming to a number of processes and events that may change the state of the world as time goes by. Both processes and events are applied as soon as their preconditions become satisfied (must transitions); differently, actions are decisions that need to be taken (may transitions). A PDDL2.1 problem is a variant where there is no time, and we only seek a sequence of actions that starts from some initial state and yields a state satisfying the goal.

Following Percassi, Scala, and Vallati (2021c) and Shin and Davis (2005), we formalise the PDDL+ semantics through the notion of time points, histories and the projection of a plan given a domain.

$\gamma(s, \cdot)$ denotes the state resulting by applying either an action/event ($\gamma(s, a)$) or a sequence of action/events ($\gamma(s, \langle a_0, \ldots, a_n \rangle)$) in state s. As for formal specification (Fox and Long 2003), an action is valid if no numeric variable appears on the left-hand side of the effect (lvalue) in more than one effect.

A time point T is a pair $\langle t, n \rangle$ where $t \in \mathbb{R}$ and $n \in \mathbb{N}$. Time points over $\mathbb{R} \times \mathbb{N}$ are ordered lexicographically. A history $\mathbb{H}$ over $\mathcal{I} = [T_s, T_e]$ 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 sequence of actions 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 in the state at time point T by $v \in F \cup X$ and by a numeric expression ξ, respectively. $E_{\mathit{trigg}}(T)$ indicates the set of active events in T. T is a significant time point of $\mathbb{H}$ over $[T_s, T_e]$ iff, in such a time point, an action is applied, an event is triggered, a process has started or stopped or there has been a discrete change just before.

The net derivative expressions are discretised using $\Delta(\xi, \delta) = \xi \cdot \delta$. For example, let $\langle x, 1.5 \cdot y \rangle$ ($\dot{x} = 1.5 \cdot y$) and $\delta = 2$ be a continuous numeric effect and a discretisation parameter, the discretised expression of x is equal to $\Delta(1.5 \cdot y, \delta) = 3 \cdot y$.

Definition 5 (Induced discrete projection). *Let* $\delta \in \mathbb{Q}$, $\mathbb{H}^\pi$ *be a history,* I *be an initial state and* π_t *be a* PDDL+ *plan. We say that* $\mathbb{H}^\pi$ *is a* discrete projection *of* π_t *which starts in* I *iff* $\mathbb{H}^\pi$ *induces the significant time points* $T_{\mathbb{H}} = \langle T_0 = \langle t_s, 0 \rangle, \cdots, T_m = \langle t_e, n_m \rangle \rangle$ *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 $E_{\mathit{trigg}}(T_i) \neq \langle\rangle$ *iff* $\mathbb{H}^\pi_s(T_{i+1}) = \gamma(\mathbb{H}^\pi_s(T_i), E_{\mathit{trigg}}(T_i))$, $\mathbb{H}^\pi_A(T_i) = \langle\rangle$, $t_{i+1} = t_i$ *and* $n_{i+1} = n_i + 1$;

R2 $\mathbb{H}^\pi_A(T_i) \neq \langle\rangle$ *iff* $\mathbb{H}^\pi_s(T_{i+1}) = \gamma(\mathbb{H}^\pi_s(T_i), \mathbb{H}^\pi_A(T_i))$, $E_{\mathit{trigg}}(T_i) = \langle\rangle$, $t_{i+1} = t_i$ *and* $n_{i+1} = n_i + 1$;

R3 *for each* $\langle a_i, t_i \rangle, \langle a_j, t_j \rangle \in \pi$, *with* $i < j$ *and* $t_i = t_j$ *there exists* $T_k, T_z \in T_{\mathbb{H}}$ *such that* $a_i \in \mathbb{H}^\pi_A(T_k)$ *and* $a_j \in \mathbb{H}^\pi_A(T_z)$ *where* $t_k = t_z = t_i$ *and* $n_k < n_z$;

R4 *for each pair of contiguous significant time points* $T_i = \langle t_i, n_i \rangle$, $T_{i+1} = \langle t_{i+1}, 0 \rangle$ *such that* $t_{i+1} = t_i + \delta$, *the value of each numeric variable* $x \in X$ *is updated as:*

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

and values of unaffected variables remain unchanged (frame-axiom).

With reference to the rules R1–R4 of the Def. 5, R1 (R2) states that if at least an action (event) is executed (triggered) in a significant time point $\langle t, n\rangle$, then there necessary exists a successor, i.e., $\langle t, n+1\rangle$, whose state associated is calculated by simply applying the discrete effects of the action(s) (event(s)). R3 is used to enforce how actions of a PDDL+ plan π are projected over a history, preserving their original ordering in case they share the same time-stamp in π. R4 is used to enforce how a numeric variable evolves when time advances for a discrete quantum of time δ.

Definition 6 (Valid PDDL+ plan under δ discretisation)**.** *π_t is a valid plan for Π under δ discretisation iff $\mathbb{H}^{\pi}_{s}(T_m) \models G$ and, for each $T \in \mathcal{I}$ such that $\mathbb{H}^{\pi}_{A}(T) \neq \langle\rangle$, then $\mathbb{H}^{\pi}_{s}(T) \models pre(a)$.*

The validity of a plan for PDDL2.1 can be formalised in a much simpler way.

Definition 7 (Valid PDDL2.1 plan)**.** *A plan $\pi = \langle a_0, ..., a_{n-1}\rangle$ is valid for a PDDL2.1 problem $\Pi = \langle F, X, I, G, A, c\rangle$ if the trajectory of states $\langle s_0 = I, s_1, ..., s_n\rangle$ generated applying actions from π iteratively is such that (i) each action a_i is executable in s_{i-1}, (ii) $s_n \models G$.*

From Discretised PDDL+ To PDDL2.1

In this section, we describe the POLY and EXP translations (Percassi, Scala, and Vallati 2021c), as well as POLY^{-}, a variant of POLY, having the characteristic of being sound but incomplete (Percassi, Scala, and Vallati 2021a,b). For the sake of clarity and readability, here we focus our attention on event-free PDDL+ tasks; the discussed translations can be straightforwardly extended to handle events (Percassi, Scala, and Vallati 2021c).

Exponential Translation

Given an event-free PDDL+ problem $\Pi = \langle F, X, I, G, A, \emptyset, P\rangle$, we define a context $\mathcal{C}$ to be a non-empty subset of processes, and denote with $\mathcal{P}^{+}(P)$ the set of non-empty subsets of P, that is the set of all possible contexts.

For an event-free PDDL+ problem Π, the exponential translation generates a PDDL2.1 problem $\Pi_{\text{EXP}} = \langle F, X, I, G, A \cup \{SIM\}, c\rangle$, discretised in $\delta \in \mathbb{Q}$. Π_{EXP} is almost identical to Π but for the absence of processes and the presence of the special action *SIM* playing the role of the simulator, i.e., what changes when time goes forward. *SIM* is defined as follows:

$$pre(SIM) = \top$$
$$eff(SIM) = \bigcup_{\mathcal{C} \in \mathcal{P}^{+}(P)} \{contpre(\mathcal{C}) \rhd conteff(\mathcal{C})\}$$

where

$$contpre(\mathcal{C}) = \bigwedge_{\rho \in P \setminus \mathcal{C}} \neg pre(\rho) \wedge \bigwedge_{\rho \in P \cap \mathcal{C}} pre(\rho)$$
$$conteff(\mathcal{C}) = \bigcup_{x \in X} \{\langle inc, x, \sum_{\substack{\langle x', \xi\rangle \in eff(\rho),\ x' = x \\ \rho \in \mathcal{C}}} \Delta(\xi, \delta)\rangle\}$$

Intuitively, the action *SIM* organises all possible contexts within a unique action, delegating to each conditional effect (i) the conditions under which a context is triggered and (ii) the consequences that such a context has on the state after some time δ has passed. Point (i) is formalised by conjoining two conjunctions: the first ensures that no other process of some other context has its precondition satisfied ($\bigwedge_{\rho \in P \setminus \mathcal{C}} \neg pre(\rho)$); the second ensures that all the preconditions of a given context are satisfied ($\bigwedge_{\rho \in P \cap \mathcal{C}} pre(\rho)$). Let x be some numeric variable of our problem, point (ii) is obtained by summing the contribution of each process within the context.

Polynomial Translation

As shown in (Percassi, Scala, and Vallati 2021c), it is possible to translate a discretised PDDL+ into a PDDL2.1 that is only polynomial with regards to the size of the PDDL+. The key idea in POLY consists in simulating the progress of a discrete amount of time $\delta \in \mathbb{Q}$ by means of a sequence of actions.

Let $\Pi = \langle F, X, I, G, A, \emptyset, P\rangle$ be an event-free PDDL+ problem, and a discretisation parameter δ, POLY generates a new PDDL2.1 problem $\Pi_{\text{POLY}} = \langle F \cup D \cup \{pause\}, X \cup X^{cp}, I, G \wedge \neg pause, A_c \cup A_P \cup \{start, end\}, c\rangle$ such that:

$$X^{cp} = \{x^{copy} \mid x \in X\}$$
$$D = \bigcup_{\substack{ne \in eff(\rho) \\ \rho \in P}} \{done_{ne}\}$$
$$A_c = \{\langle pre(a) \wedge \neg pause, eff(a)\rangle \mid a \in A\}$$
$$start = \langle \neg pause, \{pause\} \cup \bigcup_{x \in X} \{\langle asgn, x^{copy}, x\rangle\}\rangle$$
$$end = \langle \bigwedge_{done \in D} done \wedge pause, \{\neg pause\} \cup \bigcup_{done \in D} \{\neg done\}\rangle$$
$$A_P = \bigcup_{\substack{ne : \langle x, \xi\rangle \in eff(\rho) \\ \rho \in P}} \{\langle pause \wedge \neg done_{ne}, \{\sigma(pre(\rho), X^{cp}) \rhd \{\langle inc, x, \Delta(\delta, \sigma(\xi, X^{cp}))\rangle\}\} \cup \{done_{ne}\}\rangle\}$$

Whenever the passage of a discrete amount of time δ has to be simulated within Π_{POLY}, the sequence of actions $wait = \langle start, seq(A_P), end\rangle$, where $seq(A_P)$ is any permutation of all A_P actions, has to be performed. Such simulation consists of the following steps: (i) *start*, this action enables the execution of all A_P actions and, at the same time, disables all those that do not belong to A_P through the use of the *pause* predicate; (ii) $seq(A_P)$, this sequence modifies the state of the world according to the dynamics of the active processes; to prevent the A_P actions from interfering with each other, the *start* action performs a copy of all the numeric variables X, assigning the current value to the corresponding X^{cp} variables; this allows to correctly modify the state of the world, regardless of the specific sorting chosen for $seq(A_P)$; (iii) *end*, this actions closes the simulation. *end* can be executed if all the A_P actions have been executed.

Incomplete Polynomial Translation

For an event-free PDDL+ problem Π, POLY$^-$ generates a PDDL2.1 problem $\Pi_{\text{POLY}^-} = \langle F, X, I, G, A \cup \{\mathit{SIM}\}, c\rangle$, discretised in δ. Π_{POLY^-} has an similar structure to Π_{EXP} in that it also uses the *SIM* action, which is however defined differently. That is:

$$pre(\mathit{SIM}) = \top$$
$$\mathit{eff}(\mathit{SIM}) = \bigcup_{\rho \in P} \{pre(\rho) \triangleright \bigcup_{\langle x,\xi\rangle \in \mathit{eff}(\rho)} \{\langle inc, x, \Delta(\xi,\delta)\rangle\}\}$$

SIM action is always applicable and features a conditional effect for each process $\rho \in P$. Such conditional effect is triggered if the precondition of ρ holds when *SIM* is applied, modifying, for each $\langle x, \xi\rangle \in \mathit{eff}(\rho)$, the affected numeric variable x according to the discretised effect expression, i.e $\Delta(\xi, \delta)$.

According to the PDDL2.1 semantics, an action having two effects acting on the same variable is to be considered inapplicable as it would produce an undefined outcome. The *SIM* action could generate conditional effects potentially conflicting and thus become inapplicable, thus preventing, in some states, the simulation of the flow of time.

It is possible to enumerate all the states such for which *SIM* is inapplicable, i.e., the set of all states in which at least two processes affecting the same numeric variable are activated. Such a set is formally defined as follows $\mathit{FS}(\Pi_{\text{POLY}^-}) = \{s \mid s \in states(\Pi), s \models \bigvee_{\langle\rho,\rho'\rangle \in \mathit{FP}(\Pi)} pre(\rho) \wedge pre(\rho')\}$ and $\mathit{FP}(\Pi) = \{\langle\rho,\rho'\rangle \mid \langle x,\xi\rangle \in \mathit{eff}(\rho), \langle x',\xi'\rangle \in \mathit{eff}(\rho'), x = x',\ \rho \neq \rho'\}$ (where $\mathit{FS}(\cdot)$ and $\mathit{FP}(\cdot)$ stand for *forbidden states* and *forbidden pairs*, respectively).

Let π be a valid plan for Π such that $s \in \mathit{FS}(\Pi_{\text{POLY}^-})$ is traversed by π, it follows that there is no equivalent solution of π for Π_{POLY^-}. So, in the general case, POLY$^-$ is incomplete.

However, it is possible to define a syntactic property for PDDL+ problems for which it can be guaranteed that $\mathit{FS}(\Pi_{\text{POLY}^-}) = \emptyset$. Such a property, namely *1-lhs* (which stands for mono left-hand side), requires that Π does not have two processes having numeric continuous effects affecting the same numeric variable. Using this property a subclass of PDDL+ problems for which $\mathit{FS}(\Pi) = \emptyset$ is defined and therefore for which POLY$^-$ is both sound and complete. For a more in-depth discussion, the interested reader is referred to (Percassi, Scala, and Vallati 2021a,b).

Making EXP structure-sensitive

Translations for which a single action is performed to simulate the flow of a discrete amount of time, like EXP and POLY$^-$, are ideally preferable to translations where the simulation requires the execution of a sequence of actions, as in the case of POLY. What limits the practical use of EXP and POLY$^-$ is that the former, generating an exponential number of conditional effects with respect to the number of processes, is infeasible when $|P|$ is large. The second, although being very efficient for *1-lhs* PDDL+ tasks, risks making the problem unsolvable. This is particularly evident in those cases where every solution requires the activation of at least two processes affecting the same variable.

We propose a new translation, namely EXP$^{\mathcal{L}}$, having the same convenience in terms of search effort of EXP and POLY$^-$, but mitigating their negative aspects. Compared to POLY$^-$, EXP$^{\mathcal{L}}$ provides guarantees of soundness and completeness in the general case while, compared to EXP, it produces numeric tasks that are generally more feasible, since it is able to exploit the structure of the problem. Finally, for *1-lhs* tasks, EXP$^{\mathcal{L}}$ produces numeric tasks equivalent to what is produced by POLY$^-$. In what follows we formalise EXP$^{\mathcal{L}}$.

The basic idea of EXP consists in enumerating all the possible non-empty contexts $\mathcal{C}$, and, for each of them, generating a conditional effect that will be activated individually when *SIM* is applied. As it is possible to note, this approach can become quickly inapplicable with problems having a large number of processes. EXP$^{\mathcal{L}}$ overcomes this weakness by enumerating, for each numeric variable $x \in X$, all the contexts in which only the processes that affect x are considered. To present this translation we need to introduce two new definitions.

Definition 8. *Let $x \in X$ be a numeric variable. We define the following sets:* $\mathbb{E}(x) = \{\langle\xi,\rho\rangle \mid \langle x',\xi\rangle \in \mathit{eff}(\rho),\ \rho \in P, x = x'\}$ *and* $\mathbb{E}_P(x) = \{\rho \mid \langle x,\rho\rangle \in \mathbb{E}(x)\}$.

$\mathbb{E}(x)$ is the set of all the continuous numeric effects of Π affecting x together with the associated process. $\mathbb{E}_P(x)$ is the processes view of $\mathbb{E}(x)$. Let $x \in X, \mathcal{C} \subseteq \mathbb{E}_P(x)$ is what we call the *local context relevant to x*.

For an event-free PDDL+ problem $\Pi = \langle F, X, I, G, A, \emptyset, P\rangle$, EXP$^{\mathcal{L}}$ generates a PDDL2.1 problem $\Pi_{\text{EXP}^{\mathcal{L}}} = \langle F, X, I, G, A \cup \{\mathit{SIM}\}, c\rangle$, discretised in δ where:

$$pre(\mathit{SIM}) = \top$$
$$\mathit{eff}(\mathit{SIM}) = \bigcup_{\substack{x \in X,\\ \mathbb{E}(x) \neq \emptyset}} \bigcup_{\mathcal{C} \in \mathcal{P}^+(\mathbb{E}_P(x))} \{contpre(\mathcal{C}, \mathbb{E}_P(x)) \triangleright \{\langle inc, x, \sum_{\substack{\langle\xi,\rho\rangle \in \mathbb{E}(x),\\ \rho \in \mathcal{C}}} \Delta(\xi,\delta)\rangle\}\}$$
$$contpre(\mathcal{C}, P') = \bigwedge_{\rho \in P' \setminus \mathcal{C}} \neg pre(\rho) \wedge \bigwedge_{\rho \in P' \cap \mathcal{C}} pre(\rho)$$

EXP$^{\mathcal{L}}$ adds to the effects of *SIM* a set of conditional effects for each numeric variable $x \in X$ for which there is at least one process capable of affecting it, i.e. for which $\mathbb{E}(x) \neq \emptyset$ holds. Note that, for each variable, EXP$^{\mathcal{L}}$ generates a singleton with a single numeric assignment that collects the contribution of all the discretised numeric effects affecting x.

Whenever *SIM* is applied, at most one conditional effect will be activated for each numeric variable.

Proposition 1 (Soundness and Completeness of EXP$^{\mathcal{L}}$)**.** *Let $\Pi = \langle F, X, I, G, A, \emptyset, P\rangle$ be a* PDDL+ *problem, and let $\Pi_{\text{EXP}^{\mathcal{L}}} = \langle F, X, I, G, A \cup \{\mathit{SIM}\}, c\rangle$ be the* PDDL2.1 *problem obtained by using* EXP$^{\mathcal{L}}$ *translation discretised in $t = \delta$. Π admits a solution under δ discretisation iff so does $\Pi_{\text{EXP}^{\mathcal{L}}}$.*

Properties

In this section, we study the properties of the presented translations to provide a means for theoretically comparing them.

We evaluate all the schemata in terms of size of the translated numeric task, and structure of the induced search space. More precisely, given a PDDL+ task Π, we define $N_{max} = \max_{x \in X} |\mathbb{E}(x)|$ and $N_{tot} = \sum_{x \in X} |\mathbb{E}(x)|$ where $\mathbb{E}(x)$ is the set of continuous effects affecting x. N_{tot} is the number of continuous numeric effects of Π, while N_{max} is the maximum number of continuous numeric effects that could affect a numeric variable.

Let Π be a PDDL+ problem and let $Z \in \{\text{EXP}, \text{POLY}, \text{POLY}^-, \text{EXP}^{\mathcal{L}}\}$, we denote with $\Pi_Z = \langle F_Z, X_Z, I_Z, G_Z, A_Z, c_Z \rangle$ the numeric task obtained using Z.

For all $Z \in \{\text{EXP}, \text{POLY}^-, \text{EXP}^{\mathcal{L}}\}$, Z generates a numeric task Π_Z in which $|A_Z| = |A| + 1$, $|F_Z| = |F|$ and $|X_Z| = |X|$. This is due to the fact that these schemata add a single action, i.e., *SIM*, and do not add any new variable.

POLY adds an action for each continuous numeric effect of Π, and two actions to initialise and close the simulation; therefore $|A_{\text{POLY}}| = |A| + N_{tot} + 2$. Then, POLY only adds predicates *pause* and *done*, for each continuous numeric effect. Therefore, $|F_{\text{POLY}}| = |F| + |D| + 1 = |F| + N_{tot} + 1$ with D being the set of *done* predicates. Finally, to ensure that the simulation sequence $\langle start, seq(A_P), end \rangle$ produces an outcome consistent with the PDDL+ semantics, POLY doubles the numeric variables; therefore $|X_{\text{POLY}}| = 2 \cdot |X|$.

To have a clearer picture of the actual size of the problem, we also consider the number of conditional effects of Π_Z, denoted by $|\mathcal{W}_Z|$.

POLY associates to each $a \in A_P$ a conditional effect that in turns is associated with a single continuous numeric effect of Π. Therefore we get that $|\mathcal{W}_{\text{POLY}}| = \sum_{x \in X} |\mathbb{E}(x)| = N_{tot}$. POLY$^-$ associates a conditional effect for each process. Such a conditional effect is included in the effects of the *SIM* action; therefore $|\mathcal{W}_{\text{POLY}^-}| = |P|$.

EXP generates a conditional effect for each possible context $\mathcal{C} \in \mathcal{P}^+(P)$; therefore $|\mathcal{W}_{\text{EXP}}| = 2^{|P|} - 1$. As concerns EXP$^{\mathcal{L}}$, a set of conditional effects for each $x \in X$ is added to the effects of the *SIM* action. The size of each such set has cardinality equal to the set of local contexts referred to x. Therefore $|\mathcal{W}_{\text{EXP}^{\mathcal{L}}}| = \sum_{x \in X} 2^{|\mathbb{E}(x)|} - 1 = \mathcal{O}(2^{N_{max}})$.

Note that EXP$^{\mathcal{L}}$ can generate up to $2^{|P|} - 1$ whenever at least one variable is affected by all processes. Yet, as we will see in the experimental section, we have observed that this is quite a rare situation, i.e. we often observe that $2^{N_{max}} \ll 2^{|P|}$.

Table 1 provides an overview of the main theoretical properties of the translations and gives an intuition on the size of the translated planning tasks.

Comparative example

To better characterise the structure of the numeric tasks generated by the considered translations, here we describe a synthetic event-free PDDL+ problem and present the resulting PDDL2.1 tasks.

Example 1. *Let $\Pi = \langle F, X, I, G, A, \emptyset, P \rangle$ be a PDDL+ problem without events encompassing one Boolean variable, i.e., $F = \{f_1.f_2\}$, four numeric variables, i.e., $X = \{x_1, x_2, x_3, x_4\}$ and two processes $P = \{\rho_1, \rho_2, \rho_3\}$ such that: $\rho_1 = \langle x_1 > 0, \{\langle x_2, x_3 \rangle\}\rangle$, $\rho_2 = \langle f_1, \{\langle x_2, x_4 \rangle\}\rangle$ and $\rho_3 = \langle f_2, \{\langle x_1, x_3 \rangle\}\rangle$. According to R4 of Def. 5, ρ_1 affects x_2 according to $\dot{x_2} = x_3$ when $x_1 > 0$ holds, ρ_2 affects x_2 according to $\dot{x_2} = x_4$ when f_1 holds and ρ_3 affects x_1 according to $\dot{x_1} = x_3$ when f_2 hold. Finally, if $x_1 > 0 \wedge f_1$, then $\dot{x_2} = x_3 + x_4$. We enumerate the possible non-empty contexts of Π, i.e.,*

$$\mathcal{P}^+(P) = \{\{\rho_1\}, \{\rho_2\}, \{\rho_3\}, \{\rho_1, \rho_2\}, \{\rho_1, \rho_3\}, \{\rho_2, \rho_3\}, \{\rho_1, \rho_2, \rho_3\}\}$$

Using Def 8 we get that $\mathbb{E}(x_1) = \{\langle x_3, \rho_3 \rangle\}$, $\mathbb{E}(x_2) = \{\langle x_3, \rho_1 \rangle, \langle x_4, \rho_2 \rangle\}$, $\mathbb{E}(x_3) = \{\}$, $\mathbb{E}(x_4) = \{\}$, $\mathbb{E}_P(x_1) = \{\rho_1, \rho_2\}$, $\mathbb{E}_P(x_2) = \{\rho_3\}$, $\mathcal{P}^+(\mathbb{E}_P(x_1)) = \{\{\rho_1\}, \{\rho_2\}, \{\rho_1, \rho_2\}\}$ and $\mathcal{P}^+(\mathbb{E}_P(x_2)) = \{\{\rho_3\}\}$. So, $N_{tot} = 3$ and $N_{max} = 2$.

EXP *The PDDL2.1 problem obtained using EXP discretised in $t = \delta$ is $\Pi_{\text{POLY}} = \langle F, X, I, G, A \cup \{SIM\}, c \rangle$ where:*

$$SIM = \langle \top, \mathcal{W}_{\text{EXP}} = \{\mathcal{W}_{\{\rho_1\}}, \mathcal{W}_{\{\rho_2\}}, \mathcal{W}_{\{\rho_3\}}, \mathcal{W}_{\{\rho_1,\rho_2\}}, \mathcal{W}_{\{\rho_1,\rho_3\}}, \mathcal{W}_{\{\rho_2,\rho_3\}}, \mathcal{W}_{\{\rho_1,\rho_2,\rho_3\}}\}\rangle$$

$$\begin{aligned}
\mathcal{W}_{\{\rho_1\}} &= \langle x > 1 \rangle \wedge \neg f_1 \wedge \neg f_2 \triangleright \{\langle inc, x_2, x_3 \cdot \delta \rangle\} \\
\mathcal{W}_{\{\rho_2\}} &= \neg \langle x > 1 \rangle \wedge f_1 \wedge \neg f_2 \triangleright \{\langle inc, x_2, x_4 \cdot \delta \rangle\} \\
\mathcal{W}_{\{\rho_3\}} &= \neg \langle x > 1 \rangle \wedge \neg f_1 \wedge f_2 \triangleright \{\langle inc, x_1, x_3 \cdot \delta \rangle\} \\
&\dots \\
\mathcal{W}_{\{\rho_1,\rho_2,\rho_3\}} &= \langle x > 1 \rangle \wedge f_1 \wedge f_2 \triangleright \{\langle inc, x_2, (x_3 + x_4) \cdot \delta \rangle, \\
&\qquad \langle inc, x_1, x_3 \cdot \delta \rangle\}
\end{aligned}$$

SIM enumerates all the possible contexts in its conditional effects, i.e., $\mathcal{W}_{\text{EXP}} = \{\mathcal{W}_{\{\rho_1\}}, \mathcal{W}_{\{\rho_2\}}, ..., \mathcal{W}_{\{\rho_1,\rho_2,\rho_3\}}\}$, which are mutually exclusive. Whenever SIM is executed in a state, the active context triggers one and only one conditional effect of $\mathcal{W}_{\text{EXP}}$.

POLY *Let $ne_1 = \langle x_2, x_3 \rangle$, $ne_2 = \langle x_2, x_4 \rangle$ and $ne_3 = \langle x_1, x_3 \rangle$ be the numeric continuous effects of ρ_1, ρ_2, and ρ_3 respectively. The PDDL2.1 problem obtained using POLY discretised in $t = \delta$ is $\Pi_{\text{POLY}} = \langle F \cup \{done_{ne_1}, done_{ne_2}, done_{ne_3}\} \cup \{pause\}, X \cup \{x_1^{copy}, x_2^{copy}, x_3^{copy}, x_4^{copy}\}, I, G \wedge \neg pause, A_c \cup \{SIM\text{-}ne_1, SIM\text{-}ne_2, SIM\text{-}ne_3\} \cup \{start, end\}, c \rangle$ such that:*

$$\begin{aligned}
start &= \langle \neg p, \{\langle asgn, x_1^{copy}, x_1 \rangle, \langle asgn, x_2^{copy}, x_2 \rangle, \\
&\qquad \langle asgn, x_3^{copy}, x_3 \rangle, \langle asgn, x_4^{copy}, x_4 \rangle, p\}\rangle \\
SIM\text{-}ne_1 &= \langle p \wedge \neg d_{ne_1}, \{(x_1^{copy} > 0) \triangleright \{\langle inc, x_2, x_3^{copy} \cdot \delta \rangle\}, d_{ne_1}\}\rangle \\
SIM\text{-}ne_2 &= \langle p \wedge \neg d_{ne_2}, \{f_1 \triangleright \{\langle inc, x_2, x_4^{copy} \cdot \delta \rangle\}, d_{ne_2}\}\rangle \\
SIM\text{-}ne_3 &= \langle p \wedge \neg d_{ne_3}, \{f_2 \triangleright \{\langle inc, x_1, x_3^{copy} \cdot \delta \rangle\}, d_{ne_3}\}\rangle \\
end &= \langle p \wedge d_{ne_1} \wedge d_{ne_2} \wedge d_{ne_3}, \{\neg p, \neg d_{ne_1}, \neg d_{ne_2}, \neg d_{ne_3}\}\rangle
\end{aligned}$$

Note that, for reasons of space, we have shortened pause to p and done to d in the snippet above.

	POLY	EXP	POLY^{-}	EXP$^{\mathcal{L}}$
soundness	✓ Lemma 2 (⇐) in Percassi et. al, 2021c	✓ Lemma 1 (⇐) in Percassi et. al, 2021c	✓ Prop. 1 in Percassi et. al, 2021b	✓ Prop. 1 (⇐)
completeness	✓ Lemma 2 (⇒) in Percassi et. al, 2021c	✓ Lemma 1 (⇒) in Percassi et. al, 2021c	✗ in the general case Prop. 1 in Percassi et. al, 2021b ✓ if Π is *1-lhs* Prop. 3 in Percassi et. al, 2021b	✓ Prop. 1 (⇒)
$\lvert F_Z \rvert$	$\lvert F\rvert + N_{tot} + 1$	$\lvert F\rvert$	$\lvert F\rvert$	$\lvert F\rvert$
$\lvert X_Z \rvert$	$2 \cdot \lvert X\rvert$	$\lvert X\rvert$	$\lvert X\rvert$	$\lvert X\rvert$
$\lvert A_Z \rvert$	$\lvert F\rvert + N_{tot} + 1$	$\lvert A\rvert + 1$	$\lvert A\rvert + 1$	$\lvert A\rvert + 1$
$\lvert \mathcal{W}_Z \rvert$	N_{tot}	$2^{\lvert P\rvert}$	$\lvert P\rvert$	$\mathcal{O}(2^{\lvert N_{max}\rvert})$

Table 1: Properties of *soundness* and *completeness* and size of the numeric translated tasks obtained through $Z \in \{\text{EXP}, \text{POLY}, \text{POLY}^{-}, \text{EXP}^{\mathcal{L}}\}$ for a PDDL+ task Π.

The simulation of the advancement of time requires the execution of a sequence of operators, that is

$$\langle \mathit{start}, \mathit{seq}(\overbrace{\mathit{SIM}\text{-}\mathit{ne}_1, \mathit{SIM}\text{-}\mathit{ne}_2, \mathit{SIM}\text{-}\mathit{ne}_3, \mathit{SIM}\text{-}\mathit{ne}_4}^{A_P}), \mathit{end}\rangle$$

where seq$(\cdot)$ is an arbitrary permutation of the A_P operators, and the successor state after δ passes is computed incrementally. The usage of start, which makes a copy of each numeric variable, guarantees that regardless of the sequence chosen by seq$(\cdot)$ the outcome is always equivalent and consistent with the semantics of discrete PDDL+.

POLY^{-} *The* PDDL2.1 *problem obtained using* POLY^{-} *discretised in $t = \delta$ is $\Pi_{\text{POLY}^{-}} = \langle F, X, I, G, A \cup \{\mathit{SIM}\}, c\rangle$ such that:*

$$\mathit{SIM} = \langle \top, \{f_1 \triangleright \{\langle \mathit{inc}, x_2, x_4 \cdot \delta\rangle\}, f_2 \triangleright \{\langle \mathit{inc}, x_1, x_3 \cdot \delta\rangle\}, x_1 > 0 \triangleright \{\langle \mathit{inc}, x_2, x_3 \cdot \delta\rangle\}\}\rangle$$

Note that Π is not a 1-lhs because ne$_1$ and ne$_2$ affect the same variable, i.e., x_2, and so the transformation POLY^{-} *could not preserve the solution space of* Π. *So, if the state s in which SIM is executed is such that $s \models f_1 \wedge x_1 > 0$, then two conditional contradicting conditional effects affecting x_2 are triggered, causing an invalid transition. We can define the set of forbidden state for $\Pi_{\text{POLY}^{-}}$ as follows $\mathit{FS}(\Pi_{\text{POLY}^{-}}) = \{s \in \mathit{states}(\Pi),\ s \models f_1 \wedge x_1 > 0\}$*

EXP$^{\mathcal{L}}$ *The* PDDL2.1 *problem obtained using* POLY^{-} *discretised in $t = \delta$ is $\Pi_{\text{EXP}^{\mathcal{L}}} = \langle F, X, I, G, A \cup \{\mathit{SIM}\}, c\rangle$ such that SIM $= \langle \top, \mathcal{W}_{x_1} \cup \mathcal{W}_{x_2} \cup \mathcal{W}_{x_3} \cup \mathcal{W}_{x_4}\rangle$ and:*

$$\begin{aligned}\mathcal{W}_{x_1} =& \{\langle x_1 > 0\rangle \wedge \neg f_1 \triangleright \{\langle \mathit{inc}, x_2, x_3 \cdot \delta\rangle\},\\ & \neg\langle x_1 > 0\rangle \wedge f_1 \triangleright \{\langle \mathit{inc}, x_2, x_4 \cdot \delta\rangle\},\\ & \langle x_1 > 0\rangle \wedge f_1 \triangleright \{\langle \mathit{inc}, x_2, (x_3 + x_4) \cdot \delta\rangle\}\}\\ \mathcal{W}_{x_2} =& \{f_2 \triangleright \{\mathit{inc}, x_1, x_3 \cdot \delta\}\}\ \mathcal{W}_{x_3} = \{\},\ \mathcal{W}_{x_4} = \{\}\end{aligned}$$

In EXP$^{\mathcal{L}}$ *we generate a set of conditional effect for each numeric variable, i.e., $\mathcal{W}_{x_1}$, $\mathcal{W}_{x_2}$, $\mathcal{W}_{x_3}$ and $\mathcal{W}_{x_4}$. Each of these sets, enumerates all local context which can determine how the related numeric variable change as time passes. This local enumeration allows to significantly reduce the number of conditional effects from $2^{|P|} - 1 = 2^3 - 1 = 7$ to $2^{|\mathbb{E}(x_1)|} - 1 + 2^{|\mathbb{E}(x_2)|} - 1 + 2^{|\mathbb{E}(x_3)|} - 1 + 2^{|\mathbb{E}(x_4)|} - 1 = 2^2 - 1 + 2^2 - 1 + 2^0 - 1 + 2^0 - 1 = 4$, thus preserving the completeness of the translation.*

In summary, we get that $|\mathcal{W}_{\text{EXP}}| = 7$, $|\mathcal{W}_{\text{POLY}}| = 3$, $|\mathcal{W}^{-}_{\text{POLY}}| = 3$ and $|\mathcal{W}^{\mathcal{L}}_{\text{EXP}}| = 4$.

Experimental Results

Our analysis aims to corroborate the theoretical considerations from an empirical point of view by measuring the performance of all the discussed translations within an optimal search setting.

As the basis of our experiments, we used ENHSP20 (Scala et al. 2020), which allows tackling numeric planning tasks with non-linear dynamic and supports the use of customised search strategies. We consider two optimal search settings in which the admissible heuristics h^{blind} and h^{max} are used in turn, the latter in its numeric generalisation (Scala, Haslum, and Thiébaux 2016). We focus on optimal search because it tends to be more sensitive to the characteristics of the search space, and can therefore shed some light on the relative usefulness of the translations. We compare the performance achieved by the considered settings over the translated numeric problems obtained with EXP, POLY, POLY^{-} and EXP$^{\mathcal{L}}$ used with $\delta = 1$ followed by the (unchanged) translation that handles events (whose documentation is provided in (Percassi, Scala, and Vallati 2021c)). All experiments were run on an Intel Xeon Gold 6140M CPU with 2.30 GHz. For each instance, we allotted 180 seconds and limited memory to 8 GB. As benchmarks, we consider the following linear domains: SOLAR-ROVER (ROVER), LINEAR-CAR (LIN-CAR), LINEAR-GENERATOR (LIN-GEN), URBAN-TRAFFIC-CONTROL (UTC) from (Vallati et al. 2016), BAXTER from (Bertolucci et al. 2019) and OVERTAKING-CAR (OT-CAR). In addition, we also include two non-linear domains, i.e., DESCENT and HVAC. The benchmark suite and the translator are available at https://bit.ly/30gMyNW. Out of the considered benchmark domains, 4 satisfy the *1-lhs* property: ROVER, LIN-CAR, OT-CAR and HVAC. The re-

h^{blind}	Coverage				Time				Exp. Nodes (x 1000)			
Domain	POLY	POLY$^-$	EXP	EXP$^\mathcal{L}$	POLY	POLY$^-$	EXP	EXP$^\mathcal{L}$	POLY	POLY$^-$	EXP	EXP$^\mathcal{L}$
ROVER (20)	**1**	**1**	**1**	**1**	61.2	**20.3**	33.7	21.1	12012.8	**3622.3**	**3622.3**	3630.7
LIN-CAR (10)	**10**	**10**	**10**	**10**	3.3	3.0	2.9	**2.7**	199.9	**27.1**	**27.1**	**27.1**
LIN-GEN (10) ✗	**1**	**1**	**1**	**1**	3.1	3.2	3.3	**2.7**	83.9	24.7	**22.7**	23.7
BAXTER (20) ✗	4	**7**	0	6	51.3	**9.2**	—	10.4	1294.4	**122.7**	—	177.6
OT-CAR (20)	**5**	**5**	**5**	**5**	16.8	**3.7**	4.6	5.9	2643.7	**252.3**	**252.3**	**252.3**
DESCENT (20) ✗	2	0	**3**	**3**	19.6	—	**8.5**	9.6	443.6	—	**134.1**	166.0
HVAC (20)	0	0	0	0	—	—	—	—	—	—	—	—
Σ	23	24	20	**26**								
h^{max}	Coverage				Time				Exp. Nodes (x 1000)			
Domain	POLY	POLY$^-$	EXP	EXP$^\mathcal{L}$	POLY	POLY$^-$	EXP	EXP$^\mathcal{L}$	POLY	POLY$^-$	EXP	EXP$^\mathcal{L}$
ROVER (20)	**1**	**1**	**1**	**1**	121.8	28.7	54.1	**24.9**	11898.0	**3512.7**	**3512.7**	3518.8
LIN-CAR (10)	**10**	**10**	**10**	**10**	3.2	**2.9**	3.3	3.1	39.2	**25.1**	27.1	**25.1**
LIN-GEN (10) ✗	1	2	2	**3**	3.5	2.7	**2.6**	**2.6**	83.9	5.0	**2.0**	**2.0**
BAXTER (20) ✗	4	**7**	0	**7**	44.8	**6.5**	—	11.6	1041.4	**68.2**	—	161.4
OT-CAR (20)	**5**	**5**	**5**	**5**	16.2	**5.2**	5.5	5.5	1270.7	**277.6**	**277.6**	**277.6**
DESCENT (20) ✗	2	0	**3**	**3**	13.6	—	**8.7**	9.2	246.2	—	**112.1**	152.8
HVAC (20)	0	**16**	**16**	**16**	—	**3.5**	**3.5**	**3.5**	—	**6.9**	**6.9**	**6.9**
Σ	23	41	37	**45**								

Table 2: Performance achieved by h^{blind} (*upper*) h^{max} (*lower*) when run on models generated using the POLY, POLY$^-$, EXP and EXP$^\mathcal{L}$ translations with $\delta = 1$. Results are presented in terms of coverage (number of solved instances), average runtime, and average number of nodes expanded during the search process. Averages are calculated considering instances solved by all the approaches. "—" indicates that no instances can be considered for the average calculation. ✗ is used to indicate domain models that do not satisfy the *1-lhs* property. For these domains, there is no guarantee of optimality for the solution found over the POLY$^-$ models.

maining domains, i.e. LIN-GEN, UTC, BAXTER and DESCENT, do not satisfy the mentioned property.

Table 2 shows the optimal search performance on the considered benchmarks, translated with all the possible schemata when the two heuristics are used. UTC is omitted because none of the approaches is capable of solving any instance.

At a first glance it is easy to notice that (i) EXP$^\mathcal{L}$ is preferable for both heuristics in terms of coverage, and (ii) POLY, due to the numerous transitions required to make time flow, is penalised compared to all other approaches. Indeed, all approaches except POLY allow a significant coverage increase when h^{max} is used in place of h^{blind}.

It can also be noted that in all the *1-lhs* domains the performance in terms of expanded nodes is substantially equivalent between POLY$^-$, EXP and EXP$^\mathcal{L}$.

Considering h^{max}, EXP performs well in terms of coverage in almost all the domains due to the fact that they all have on average few processes, except for BAXTER, which has on average about 56 processes. Indeed, EXP does not solve any instance on this domain. EXP$^\mathcal{L}$ manages to handle this domain quite well. BAXTER has in fact the following structure: $\mu(N_{max}) = 9$, $min(N_{max}) = 6$ and $max(N_{max}) = 12$.

Notably, POLY$^-$ allows achieving good coverage performance when h^{max} is used. With the exception of DESCENT, where no instance is solved, the incompleteness of the translation did not turn out to be a problem in terms of solvability. It is worth reminding that, using POLY$^-$ on non *1-lhs* tasks does not guarantee the optimality of the solutions, as some solutions could be removed from the solution space due to the incompleteness of this translation. Therefore the comparison of performance POLY$^-$ and the other considered translations should be considered only indicative.

Overall, the results seem to support the theoretical considerations. In domains characterised by few processes, *SIM*-based methods are preferable to POLY and are roughly equivalent. Conversely, when the PDDL+ tasks include numerous processes, as in BAXTER, EXP becomes infeasible while EXP$^\mathcal{L}$ allows delivering the best performance. Finally, the incompleteness of POLY$^-$, although often advantageous in terms of speedup, may lead to cases where all the solutions are pruned from the search space, as in DESCENT.

Discussion

In this work, we studied a range of existing translations from discrete PDDL+ to numeric PDDL2.1 in order to understand their characteristics and to provide a means to compare them. This analysis led to the design of a new translation, namely EXP$^\mathcal{L}$, which aims to be a synthesis of the strengths of the others.

The EXP translation is a baseline with more didactics than practical purposes, but it has the advantage of not significantly lengthening the numeric plans with respect to the corresponding PDDL+ plans. However, always enumerating all the possible global contexts has the obvious limitation of being infeasible for PDDL+ problems involving many processes.

On the contrary, POLY, in which the time flow involves the execution of a polynomial number of actions (with respect to a constant of the task, i.e., N_{tot}) leads to a greater search effort which is however compensated by a greater coverage (at least in the sub-optimal context as shown in (Percassi, Scala, and Vallati 2021c)).

POLY$^-$ produces tasks having the same structure as those produced by EXP, having the same advantage in terms of search compared to POLY, but its incompleteness allows it to be used safely only for those tasks that satisfy the syntactic *1-lhs* property, which is quite restrictive. However, POLY$^-$ hinted that knowing the characteristics of the problem at hand can lead to extremely efficient translations.

The novel EXP$^{\mathcal{L}}$ has the same advantages as EXP and POLY$^-$ but, differently from EXP, which is always exponential with respect to the $|P|$ regardless of the structure of the problem considered, it generates numeric tasks which are exponential with respect to a constant of the problem, i.e. N_{max}. Since we assume that the worst-case for EXP$^{\mathcal{L}}$ in practical domains, i.e. all the processes affect all the numeric variables, is extremely rare then is it likely that in general $N_{max} \ll N_{tot}$ holds when P is large. It follows, that statistically, EXP$^{\mathcal{L}}$ will generate more feasible problems than its structure-insensitive EXP counterpart. However, we are aware that in some contexts N_{max} could be very large and EXP$^{\mathcal{L}}$ may not be more preferable than POLY.

We see several avenues for future work. We are interested in exploring the online selection of the best translation to be used according to the structure of the problem considered. We are also interested in investigating the possibility of automatically combining different translations, to further adapt to the structure of the problem at hand.

References

Bertolucci, R.; Capitanelli, A.; Maratea, M.; Mastrogiovanni, F.; and Vallati, M. 2019. Automated Planning Encodings for the Manipulation of Articulated Objects in 3D with Gravity. In *Proc. of AI*IA 2019*, 135–150.

Bonassi, L.; Gerevini, A. E.; Percassi, F.; and Scala, E. 2021. On Planning with Qualitative State-Trajectory Constraints in PDDL3 by Compiling them Away. In *Proc. of ICAPS 2021*, 46–50.

Cooper, M. C.; Maris, F.; and Régnier, P. 2010. Compilation of a High-level Temporal Planning Language into PDDL 2.1. In *Proc. of ICTAI 2010*, 181–188.

Fox, M.; and Long, D. 2003. PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains. *JAIR*, 20: 61–124.

Fox, M.; and Long, D. 2006. Modelling Mixed Discrete-Continuous Domains for Planning. *JAIR*, 27: 235–297.

Grastien, A.; and Scala, E. 2017. Intelligent Belief State Sampling for Conformant Planning. In *Proc. of IJCAI 2017*, 4317–4323.

Keyder, E.; and Geffner, H. 2009. Soft Goals Can Be Compiled Away. *JAIR*, 36: 547–556.

Palacios, H.; and Geffner, H. 2009. Compiling Uncertainty Away in Conformant Planning Problems with Bounded Width. *JAIR*, 35: 623–675.

Percassi, F.; and Gerevini, A. E. 2019. On Compiling Away PDDL3 Soft Trajectory Constraints without Using Automata. In *Proc. of ICAPS 2019*, 320–328.

Percassi, F.; Scala, E.; and Vallati, M. 2021a. A Sound (but Incomplete) Polynomial Translation from Discretised PDDL+ to Numeric Planning. In *KEPS 2021*.

Percassi, F.; Scala, E.; and Vallati, M. 2021b. A Sound (but Incomplete) Polynomial Translation from Discretised PDDL+ to Numeric Planning. In *Proc. of AI*IA 2021*.

Percassi, F.; Scala, E.; and Vallati, M. 2021c. Translations from Discretised PDDL+ to Numeric Planning. In *Proc. of ICAPS 2021*, 252–261.

Scala, E.; and Grastien, A. 2021. Non-Deterministic Conformant Planning Using a Counterexample-Guided Incremental Compilation to Classical Planning. In *Proc. of ICAPS 2021*, 299–307.

Scala, E.; Haslum, P.; and Thiébaux, S. 2016. Heuristics for Numeric Planning via Subgoaling. In *Proc. of IJCAI 2016*, 3228–3234.

Scala, E.; Haslum, P.; Thiébaux, S.; and Ramírez, M. 2020. Subgoaling Techniques for Satisficing and Optimal Numeric Planning. *JAIR*, 68: 691–752.

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

Taig, R.; and Brafman, R. I. 2013. Compiling Conformant Probabilistic Planning Problems into Classical Planning. In *Proc. of ICAPS 2013*, 197–205.

Vallati, M.; Magazzeni, D.; Schutter, B. D.; Chrpa, L.; and McCluskey, T. L. 2016. Efficient Macroscopic Urban Traffic Models for Reducing Congestion: A PDDL+ Planning Approach. In *Proc. of AAAI 2016*, 3188–3194.