The Impact of PDDL+ Language Features on Planning Performance: An Empirical Analysis on a Real-world Case Study

Anas El Kouaiti
Università degli Studi di Brescia
Brescia, Italy
a.elkouaiti@studenti.unibs.it

Francesco Percassi
University of Huddersfield
Huddersfield, UK
f.percassi@hud.ac.uk

Alessandro Saetti
Università degli Studi di Brescia
Brescia, Italy
alessandro.saetti@unibs.it

Mauro Vallati
University of Huddersfield
Huddersfield, UK
m.vallati@hud.ac.uk

Abstract

PDDL+ is an expressive formalism that allows for the use of planning in complex real-world applications. It includes a number of features designed to improve the readability and conciseness of the resulting knowledge models, but that are commonly doubted to have detrimental impact on the performance of domain-independent searches and heuristics. In this paper we empirically assess the impact of such features in a challenging real-world case study.

Introduction

Automated planning is a prominent Artificial Intelligence challenge, which is concerned with the problem of finding a sequence of actions that can bring the agent into some goal state from a given initial condition. Real-world applications often require the ability to accurately represent aspects of the environment. In response to this need, the PDDL+ language was developed to facilitate the concise encoding of hybrid models for automated planning (Fox and Long 2006). Notably, PDDL+ models are amongst the most advanced models of systems and the resulting problems are notoriously difficult for domain-independent planning engines to cope with. Complexity can be exacerbated by the use of language features that have been designed to improve readability and maintenance for knowledge engineers, but that have the potential to make the search space more difficult to explore. Considering less expressive languages from the PDDL family, there is indeed a wealth of work that focuses on reformulating knowledge models by removing the use of some poorly supported language features (Helmert 2009; Ceriani and Gerevini 2015; Percassi and Gerevini 2019).

With the aim of supporting the knowledge engineering process of PDDL+ models, in this paper we empirically assess the impact of some challenging language features on a range of domain-independent search and heuristic techniques. This assessment is carried out by considering a real-world application of planning in urban traffic control and utilising historical data for analysis. The focus for features is on *numerical assignments* and *conditional effects*.

The PDDL+ Language

A PDDL+ planning problem is formally defined by a tuple $\Pi = \langle \mathcal{F}, \mathcal{X}, \mathcal{I}, \mathcal{G}, \mathcal{A}, \mathcal{E}, \mathcal{P} \rangle$ in which each element is detailed as follows. $\mathcal{F}$ and $\mathcal{X}$ are sets of Boolean and numeric variables, respectively; the domain of a Boolean variable is $\mathbb{B} = \{\top, \bot\}$ where $\top$ and $\bot$ are the logical true and false, respectively; the domain of numeric variable is $\mathbb{Q}$. $\mathcal{I}$ is the description of the initial state, expressed as a full assignment to all variables in $\mathcal{X}$ and $\mathcal{F}$. $\mathcal{G}$ is the description of the goal, expressed as a formula. $\mathcal{A}$ and $\mathcal{E}$ are the sets of actions and events, respectively, sharing the same syntax. An action or event is a pair $\langle p, e \rangle$, where p is a propositional formula using standard connectives from logic involving numeric and Boolean conditions, and e is a set of Boolean or numeric effects. $\mathcal{P}$ is a set of processes, and a process is a pair $\langle p, e' \rangle$, where p is a propositional formula involving numeric and Boolean conditions, and e' is a set of continuous numeric effects expressed as pairs $\langle x, \xi \rangle$, where $x \in \mathcal{X}$ and ξ is a numeric expression redefining the value of x.

Conditional Effects are an expressive PDDL language feature utilised for defining *state-dependent* effects in the action model. In essence, a conditional effect of an action represents an effect that occurs only when an additional condition holds at the time when the action is applied. Widely employed in complicated scenarios, conditional effects serve as a valuable tool for compactly representing complex application domains. In a PDDL action a conditional effect is specified using the keyword `when`, and the effect (`eff`) takes place if the condition (`cond`) holds when the action is applied. Otherwise, the conditional effect is ignored.

Numerical assignments is a language feature introduced to support numeric reasoning in PDDL. It is a statement that is defined as an effect of an action model, to indicate that as a result of the action execution, a numeric variable is changing its value to a new one. An example of an assignment is `(assign (numVar) 3.0)`, indicating that `numVar` value is set to 3.0 in the state resulting from the action execution.

Case Study

In this work, we perform our empirical analysis on a version of the models presented in El Kouaiti et al. (2024), namely VARE, that extends the PDDL+ models introduced by McCluskey and Vallati (2017) to address traffic signal optimisation through automated planning. The underlying

idea is that the planning system is in charge of optimising traffic lights for portion of an urban traffic network, to achieve a predefined goal such as decongesting a link or maximising the number of vehicles leaving the area. The complete model is provided here: https://github.com/anas-elkouaiti/utc-models-deployable

Language Features and Compilations. In the considered model, assignments are used to reset to 0 numeric variables or to update numeric values due to some changes in the configuration. The first case can be compiled away by substituting the assignment effect with a subtraction of the numeric value by itself, such as `(decrease (numVar) (numVar))`.

The other assignment case refers to the update of a numeric value, that in the considered model is used in the *changeLimit* action. In this case, the reformulation requires to explicitly specify in the initial state the allowed *limit* values, e.g., 4, introduce an additional proposition `(activelimit ?j ?l)`, and modify the *changeLimit* action so that a limit `?l` can be activated on a considered junction `?j`. This requires also the modification of the parameters list of the action, that now needs to include both the currently active limit `?l1` and the limit to be assigned `?l2`.

We can now turn our attention to conditional effects, used in the *trigger-change* event. The first conditional effect, i.e., `(when (endcycle ?i ?p1) (increase (countcycle ?i) 1))`, is employed to increment the variable that keeps track of how many times the configuration, currently selected for junction j, has been executed. The second conditional effect, i.e., `(when (endcycle ?i ?p) (not (configurable ?i ?p))`, not only narrows down the search space but also maintains the integrity of the problem's correctness constraints, preventing the *changeLimit* action in invalid stages.

For the reformulation of this language feature, we followed Nebel (2000), i.e. we multiply out the *trigger-change* event, according to all the possible combinations of conditional effects. This leads to 3 events: the original *trigger-change* with no conditional effects, and a new one event for each potential branch of the starting conditional effect.

Experimental Analysis. We use the same benchmarks proposed by El Kouaiti et al. (2024). The modelled urban network area is situated in West Yorkshire, UK, and seven different traffic scenarios are considered, with five different goals each. Experiments were run on a machine with a 2.3 GHz Intel Xeon Gold 6140M CPU and 8 GB of RAM. As planning engine, we use ENHPS (Scala et al. 2020a) v20. It implements a large number of heuristics and search techniques, providing the ideal testbed. The considered search strategies are greedy best-first search (GBFS) and A*, and the adopted heuristics are h^{add} (Scala et al. 2016), h^{max} (Scala et al. 2016), and h^{mrp} (Scala et al. 2020b). These are state-of-the-art approaches in hybrid planning. We also considered blind search and A*, but no problem was solved.

Table 1 provides an overview of the results in terms of the number of solved instances, and IPC score for the quality of generated plans, i.e., their durations, and runtime. The IPC score is calculated as in IPC 2014 (Vallati, Chrpa, and McCluskey 2018): higher scores indicate higher performance.

	h^{max}	h^{mrp}	h^{add}	Σ (105)
Number of Solved Instances				
B (35)	35	34	1	70
-ce	35	34	1	70
-asgn	34	31	0	65
-ce -asgn	30	25	0	55
Σ (140)	**134**	124	2	
Quality Score				
B (35)	32.91	32.62	1.00	**66.53**
-ce	32.94	32.49	1.00	65.43
-asgn	31.99	29.62	0.00	61.61
-ce -asgn	22.09	18.34	0.00	40.43
Σ (140)	**119.93**	113.07	2.00	
Runtime Score				
B (35)	18.42	15.02	0.12	**33.56**
-ce	18.69	14.66	0.16	33.51
-asgn	13.77	10.20	0.00	23.97
-ce -asgn	6.78	5.07	0.00	11.85
Σ (140)	**57.66**	44.96	0.28	

Table 1: Coverage, quality and runtime score for each model. "B" denotes the basic formulation while *-ce* and *-asgn*, denotes, respectively, a variant where conditional effects and/or numeric assignments are compiled away.

The use of conditional effects and assignments does not harm planning performance. This is a very surprising result, as it contradicts the common knowledge that considers such features detrimental to planning performance. This is highlighted by the poor performance, according to all of the considered metrics, of the model when both conditional effects and assignments are compiled away. In terms of coverage, numeric assignments are the features that is mostly beneficial, while conditional effects do not appear to have any remarkable impact. A similar figure can be drawn when looking at quality scores and runtime.

Summarising, our extensive experimental analysis disproofs the common belief that the use of conditional effects and numeric assignments has a detrimental impact on PDDL+ search and heuristic techniques. On the contrary, we showed that their use can be beneficial and hence should not be excluded a priori. While we acknowledge that the analysis considers a single domain model, it is worth highlighting that the model is amongst the most complex benchmarks in PDDL+ planning in terms of dynamics of the environment and size of the models, hence it is the most suitable to emphasise performance differences.

Acknowledgements

Francesco Percassi and Mauro Vallati were supported by a UKRI Future Leaders Fellowship [grant number MR/T041196/1].

References

Ceriani, L., and Gerevini, A. E. 2015. Planning with always preferences by compilation into strips with action costs. In *Eighth Annual Symposium on Combinatorial Search*.

El Kouaiti, A.; Percassi, F.; Saetti, A.; McCluskey, L.; and Vallati, M. 2024. Pddl+ models for deployable yet effective traffic signal optimisation. In *34th International Conference on Automated Planning and Scheduling*.

Fox, M., and Long, D. 2006. Modelling mixed discrete-continuous domains for planning. *J. Artif. Intell. Res.* 27:235–297.

Helmert, M. 2009. Concise finite-domain representations for PDDL planning tasks. *Artif. Intell.* 173(5-6):503–535.

McCluskey, T. L., and Vallati, M. 2017. Embedding automated planning within urban traffic management operations. In *Proceedings of the Twenty-Seventh International Conference on Automated Planning and Scheduling, ICAPS*, 391–399.

Nebel, B. 2000. On the compilability and expressive power of propositional planning formalisms. *J. Artif. Intell. Res.* 12:271–315.

Percassi, F., and Gerevini, A. E. 2019. On compiling away PDDL3 soft trajectory constraints without using automata. In *Proceedings of the Twenty-Ninth International Conference on Automated Planning and Scheduling, ICAPS*, 320–328.

Scala, E.; Haslum, P.; Thiébaux, S.; and Ramirez, M. 2016. Interval-based relaxation for general numeric planning. In *ECAI 2016*, 655–663.

Scala, E.; Haslum, P.; Thiébaux, S.; and Ramirez, M. 2020a. Subgoaling Techniques for Satisficing and Optimal NumericPlanning. *J. Artif. Intell. Res.* 68:691–752.

Scala, E.; Saetti, A.; Serina, I.; and Gerevini, A. E. 2020b. Search-Guidance Mechanisms for Numeric Planning Through Subgoaling Relaxation. In *Proc. of ICAPS 2020*, 226–234. AAAI Press.

Vallati, M.; Chrpa, L.; and McCluskey, T. L. 2018. What you always wanted to know about the deterministic part of the international planning competition (IPC) 2014 (but were too afraid to ask). *Knowledge Eng. Review* 33:e3.