Planning Tech for Planning Pedagogy

Alex Coulter*, Teo Ilie*, Renee Tibando*, Christian Muise

Queen's University, Kingston, On, Canada
{alex.coulter, 17ti5, 17rat3, christian.muise}@queensu.ca

Abstract

We demonstrate the power planning techniques can have for the task of analyzing planning solutions in a classroom setting. Using the common assignment strategy of asking students to develop PDDL given an English description of a domain, we consider how a variety of planning methods (existing and new) can provide analytic support for teaching staff to understand which errors were made in student models. The work has already had a direct and practical impact, being deployed in a classroom setting to assess the correctness of student-authored planning models.

Demo: mulab.ai/demo/p4pp

1 Introduction

We consider what planning techniques can be brought to bear on the task of analyzing student-authored PDDL models in a classroom setting. We consider two existing methods – testing solutions and cross-model validation – and one novel technique for model alignment. For the analysis, we assume that a reference model is provided, and the two planning theories (reference and student's) share a number of elements from the high-level PDDL specification: types, objects, constants, and action names + parameters.

Planners are run on the student-submitted domain and problem files, and solutions are inspected for correctness. This is a baseline technique that has been commonly used for analyzing student PDDL. The next phase, making use of a reference solution, aims to validate the plans generated for one model on the other model's theory. For example, plans generated using the student submission will be validated using the reference model (and vice versa). This is meaningful, as the actions and parameters are presumed to be the same.

Finally, the newly introduced aspect of theory alignment is an encoding approach that captures if two theories are a *regular bisimulation* (Milner 1990). The area of bisimulation has been mined by the planning community for several aspects. Most notably, it has led to the extremely impressive and popular line of merge-and-shrink heuristics (Katz, Hoffmann, and Helmert 2012; Helmert et al. 2014) found in the state-of-the-art Fast Downward planning system (Helmert 2006). Loosely defined, a bisimulation is a correspondence between two transition systems such that labelled transitions between the two systems coincide: if two nodes (one from each system) coincide, then the reachable nodes in each system also coincide (following the labelled transitions). In regular bisimulation the labels of the transitions are presumed to be the same, as well as the reachable state space for each of the models (though the fluent descriptions may differ).

Due to the ultimate application of comparing PDDL models, our approach is grounded firmly in the manipulation of PDDL. Taking two planning models (represented in domain+problem PDDL files), we *merge* them through a novel encoding to a new domain+problem file. This merged domain retains the original types, action names/parameters, objects, and constants (assumed to be the equivalent in both of the original models), and combines the action specifications to progress through both models simultaneously. Similarly, fluents and initial states are merged. Finally, "failure actions" are introduced that represent a potential misalignment between the two models. As long as one of these failure actions can be executed, the two theories do not align.

The encoding was implemented and embedded as a core analytic in an undergraduate AI class where one of the assignments was to create a planning model (fluents, action preconditions/effects, initial/goal states) according to a text-based specification. Two reference models were created to compare against, and the proposed alignment was an integral part of the teaching staff's analysis of student submissions. We found that our proposed alignment was not only viable, with many submissions having "solutions" to the merged model showing where a modelling error occurs, but several cases demonstrated errors with the submitted domains that were subtle and detected only by this added approach.

2 Approach

Plans & Validations The first aspect of analysis of a student submission is (1) running their planners on their submitted PDDL to see if plausible plans are generated; and (2) running cross (planning-based) validation with solutions and the reference model. Step (1) simply involves checking if a plan is found and looks plausible. While this step is informal, deeper errors are detected through subsequent steps.

Given a student model and reference model, plans are generated for each. Step (2) involves validating (using VAL (Howey, Long, and Fox 2004)) the plans for one model with the other model. This is viable since the objects, actions,

*These authors contributed equally.

Figure 1: Single Problem Analysis

and their parameters are all presumed to be the same (fluents may differ, but these are not required for validating a plan). If validation fails, the nature of the failure often indicates where the model has gone wrong.

Finally, the deepest form of analysis is conducted by (3) attempting to *align* the two models.

Model Alignment Model alignment is conducted by combining the encodings of the two models (renaming fluents so they remain distinct). Merged actions will progress the state of the world in both models simultaneously and have the preconditions merged as well. Finally, new "failure actions" are introduced that allow the progression in one model but not the other. These failure actions add the new goal fluent `(failed)` and having a plan that executes one of them indicates that a state was found where one model diverges from the other in the actions applicable. If no plan exists on this combined model, then the theories align and the student's PDDL model is correct.

The full details of the theory are beyond the scope of this demo abstract, but further details can be found in the tech paper associated with the project.[1]

3 Example Application

As an example error detected, here we show the analysis of a single submission. Figure 1 shows the information presented to the grader. Failures for "Aligns Orig" can be seen in the following example plan from the merged domain:

```
Mis-alignment plan for p01:
(move loc12 loc22 c1222)
(pick-up loc22 key1)
(move loc22 loc23 c2223)
(unlock loc23 c2324 red key1)
(fail_unlock2 loc23 c2324 red key1)
; cost = 5 (unit cost)
```

Upon inspecting the suspicious unlock example, it is clear that the team neglected to change the locked status of the corridor (it should be deleted as an effect):

```
(:action unlock
   :parameters (...)
   :precondition (and
        ...
        (cor-locked ?cor ?col))
   :effect (and
        (cor-unlocked ?cor)
        (when (key-two-use ?k)
              (key-one-use ?k))
        (when (key-one-use ?k)
              (key-used-up ?k))))
```

This is a common modelling error and one that does not affect any of the found plans for the setting. It is only through the alignment that such errors become readily detectable.

4 Summary

We have introduced an array of planning techniques dedicated to evaluating the correctness of PDDL models given a pre-existing reference model. The deepest analysis, done by aligning the two models, is capable of fully analyzing the reachable state space of both models to ensure they capture the same environment. This provides a powerful analysis in the classroom setting for assessing the correctness of student-authored PDDL, and has already been validated through the use of the methods in grading an assignment in an undergraduate AI course at Queen's University. Preliminary evidence suggests that 3x errors were found by using planning validation techniques compared to just analyzing plans, and 6x errors were discovered by using the more advanced model alignment. These promising results exemplify the strength planning techniques can have in the area of planning pedagogy directly. Extensions currently under consideration include (1) analyzing and surfacing the causal reason for plans to exist in the aligned theory (rather than leaving it to the marker to figure out); (2) incorporating an action for the problem goals so they are considered in the alignment; (3) computing a diversity of failed plans to demonstrate multiple errors; and (4) enabling iterative model refinement so multiple errors can be detected.

References

Helmert, M. 2006. The Fast Downward Planning System. *J. Artif. Intell. Res.*, 26: 191–246.

Helmert, M.; Haslum, P.; Hoffmann, J.; and Nissim, R. 2014. Merge-and-Shrink Abstraction: A Method for Generating Lower Bounds in Factored State Spaces. *J. ACM*, 61(3): 16:1–16:63.

Howey, R.; Long, D.; and Fox, M. 2004. VAL: Automatic plan validation, continuous effects and mixed initiative planning using PDDL. In *16th IEEE International Conference on Tools with Artificial Intelligence*, 294–301. IEEE.

Katz, M.; Hoffmann, J.; and Helmert, M. 2012. How to Relax a Bisimulation? In McCluskey, L.; Williams, B. C.; Silva, J. R.; and Bonet, B., eds., *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling, ICAPS*. AAAI.

Milner, R. 1990. Operational and Algebraic Semantics of Concurrent Processes. In van Leeuwen, J., ed., *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics*, 1201–1242. Elsevier and MIT Press.

[1] To be released during the in the camera-ready version.