# Reasoning about Planning Domains

**Rajesh Kalyanam** and **Tanji Hu** and **Robert Givan**
Electrical and Computer Engineering, Purdue University, W. Lafayette, IN 47907
{rkalyana, hut, givan}@purdue.edu

**Abstract**

Humans exhibit a significant ability to generate concise generalized plans without the need to consciously consider specific problem instances. Closely related is the human ability to soundly and effectively answer interesting questions about previously unseen planning domains. Our ultimate goal is to develop the representations and reasoning methods necessary for a planner to demonstrate and utilize similar understanding of typical benchmark planning domains.

For our initial work, we focus on designing a system that can answer rich questions about simple planning domains using deductive reasoning. In this paper we discuss examples of such questions, many of which can be formulated without reference to any particular instance of the planning domain being considered. We then briefly review the reasoning and representation background we hope to exploit in building a system to answer such questions about planning domains.

## Introduction

Considering typical benchmark planning domains, humans are generally able to "understand" the domain and answer interesting questions about the domain. For instance, "Can a cycle of blocks be created?" Or, "Can we get every block onto the table?" Or, "Can we leave currently correct block towers undisturbed?" In some cases, the knowledge demonstrated in this fashion appears to enable the checking (or even construction) of generalized plans for specific abstract goals without the apparent consideration of any individual problem instances. For example, "Put all the blocks onto the table except those that are already in currently correct towers, and then build the desired towers from the bottom up" can be seen upon examination to be a correct plan for the blocks world.

Some past planning algorithms have used human provided abstract domain information to increase planning effectiveness. For instance, in TLPlan (Bacchus and Kabanza 2000) temporal logic formulas characterizing invariants of high-quality solution trajectories are used to effectively restrict the solution plans considered by an otherwise very simple search-based planner. We are interested in the reasoning process by which such invariants are found, or even by which such invariants can be checked as correct.

We adopt the perspective that deduction plays a key role in each of these human abilities. In checking the correctness of generalized plans, of course, humans need to use deduction to ensure arbitrary initial states will be carried to their desired goal states by the plan. In the case of a person writing the TLPlan pruning rules, deduction is required to ensure the soundness of the pruning rules, i.e., all problems in the domain should still be solvable after the pruning.

Our goal is to develop the representations and reasoning methods required to make such inference possible when the planner encounters a new planning domain. However, we intend some restrictions limiting our goals as well as our methods. First, we insist that each reasoning question is answered (or left unanswered) after only polynomially many reasoning steps. Second, rather than achieve tractability by limiting what can be asked, we insist on an expressive and general-purpose language for representing statements about planning domains. These two restrictions imply immediately that our question-answering ability will be logically incomplete, unable to confirm statements that are in fact entailed. Intuitively, we can think of our proposed system as a cognitive model for the human ability to draw "obvious" conclusions from reading planning domain definitions (McAllester 1991).

A trivial instance of our system would simply say "I don't know" when asked anything about a planning domain. Of course, this would be of no use in constructing plans. There is a dimension of variation between such a trivial system and a very advanced system that successfully polynomial-time verifies a wide range of generalized plans and confirms many non-trivial assertions about domains. Our goal is simply to advance along this dimension.

In this paper, we will not discuss our preliminary approaches to these questions, except to point to the reasoning work we are building upon. We design domain representations based upon taxonomic syntax for predicate logic (McAllester and Givan 1993), a variant of predicate logic developed to facilitate polynomial-time class-based inference (similar in spirit and many details to the more well known "description logic"). We build our reasoning methods on the higher-order reasoning system Ontic (McAllester

1989) that heavily leverages forward-chaining polynomial-time inference procedures to build a cognitive model for verifying mathematical arguments. We represent planning domains using a taxonomic class-based variant of the situation calculus suitable for effective reasoning in Ontic. We seek to enrich the reasoning principles in Ontic to deal well with the particular aspects of reasoning about dynamic systems; Ontic was originally designed for focus on mathematical reasoning. In particular, we build into Ontic reasoning methods that carry out automated mathematical induction to reason about the transitive closures involved in reachability between states (and within states in cases like above/on in the blocks world).

Exciting recent work exists that does address some reasoning questions about planning domains (Srivastava, Immerman, and Zilberstein 2010), in particular automatically checking termination and correctness for interesting classes of generalized plans. We seek to increase the flexibility and depth of semantic representation beyond that pre-existing work using a substantially different approach.

## Reasoning Challenges

Here we sketch simple variants on familiar planning domains and simple questions about these variants that we claim people can easily address. The intent is to inspire work to build flexible systems that can answer and/or verify the answers to these questions. Some of the queries below would be far beyond the current state of the art in reasoning, given the goal of answering the query without any further human assistance. However, most of these queries can be answered by reasoning systems given sufficient human assistance (e.g. proof text). Our research endeavor, then, is to reduce the necessary human assistance by strengthening the built-in domain-independent reasoning about dynamic systems in the direction of the reasoning humans are able to perform apparently effortlessly.

**Blocks World** Of course, the simple traditional blocks world already contains a number of examples of observations that are trivial to humans but difficult for reasoners. We start by listing a few of these. We note that, as easy as these conclusions are for humans to verify with high confidence, a formal proof may be quite sophisticated and will typically include rich representation and mathematical induction.

- A desired `on` fact may need to be temporarily destroyed.
- Any particular `on` fact can be achieved while preserving all current `on` facts except those 'above' the blocks involved.
- Any block can be made `clear` while preserving all other `clear` facts.
- Any block can be put on the table while preserving the set of blocks that are already on the table.

**Space-restricted Blocks World** Here, we consider the blocks world where the table has room for only $n$ blocks.

- When $n = 2$, all reachable states are partitions of the initial state into two towers.
- When $n = 3$, all states with three or fewer towers are reachable.

The universal reachability when $n = 3$ is not immediately obvious, and a suitable reasoning goal would be to have a system capable of checking this fact with some additional human guidance, i.e., checking a concise proof of this fact or checking the correctness of a concise generalized plan.

**Modified-pickup Blocks World** In this domain variant, the pickup action is modified so that it can pick up towers of height more than one (even if not on the table) but cannot pick up individual blocks unless they are both clear and on the table.

- Starting from a state where there is atleast one tower of height greater than one, the state where all the blocks are on the table is not reachable.

**Colored Blocks World** Other interesting questions can be posed in a colored blocks world variant in which we have blocks of two different colors, red and green. This variant restricts block stacking (and the initial state) so that no block may be stacked on another block of the same color. The following facts about all reachable states are typically easily verified by humans.

- Any tower has at most one more red block than green block (and vice versa).
- If there are $k$ more green blocks than red blocks, then there are at least $k$ `clear` green blocks.

**Logistics** We also provide examples from the directed and undirected variants of the traditional logistics domain with no resource constraints. Here, we discuss the problem with one city, many locations and trucks within that city, and no airplanes.

- If there are more packages than there are trucks, then some truck must deliver more than one package.
- No package can be both on a truck and at a location different from the truck's location. Invariants like this have been studied in (Fox and Long 1998).
- In undirected-graph logistics:
  - Unloading can be restricted to package destinations.
  - If there is no path between two packages, then the same truck cannot load both packages, even at different times.
- The following are necessary conditions for solvability in directed-graph logistics:
  - The destination of each package is reachable from its source
  - Some truck can reach the source location for each package

- The above two conditions are not sufficient for solvability. Consider two packages requiring the same truck to take different irreversible drives.
- In directed logistics, unloading cannot be restricted to package destinations. Consider a truck waiting at a fork between irreversible steps, and another truck arriving with two packages needing to travel different branches of the fork.

**Pipesworld** The pipesworld domain is one of the deterministic planning competition benchmarks (Hoffmann and Edelkamp 2005). It is a subtle variant of the traditional logistics domain with pipes connecting locations, referred to as "areas", across which individual packages, referred to as "batch atoms", can be transported by pushing them into pipes. A batch atom is transported through a pipe by pushing additional batch atoms in behind it until it emerges at a new area. Interface restrictions among the batch atoms restrict the pairs of batch atoms that can be adjacent in any pipe.[1]

- The number of atoms in a pipe never changes. A corollary of this fact is that no area can ever contain all batch atoms.
- If a circular pipe network exists between a set of areas, then any batch atom can be moved from any area on this network to any other (as long as there are no interface restrictions). Further properties are also apparent about the nature of interface restrictions that would impact this claim.
- If there are two kinds of batch atoms: red and green, and red batch atoms cannot be adjacent to green batch items, then:
  - All atoms in any one pipe are the same color. Thus we can refer to the "color" of a pipe.
  - The "color" of a pipe is unchangeable.
  - The number of atoms of each color outside of pipes cannot change.
  - No area can ever contain all the red batch atoms unless it does so in the initial state.

## Conclusion

In this paper we discussed a small set of example questions that humans are able to effectively answer when provided with a planning domain. We believe that any sufficiently general reasoning system that is capable of answering such questions can also aid in the construction and verification of generalized plans. Our immediate goal is to develop a reasoning system able to verify rich state invariants for simple planning domains such as the blocks world (e.g., that no cycle of blocks can be formed). The inference of state invariants is of course a particular instance of the question-answering paradigm that we described above. Our eventual goal however, is to apply this reasoning system to the richer range of questions discussed above. Intermediate subgoals exist in which question-answering or state-invariant verification can be conducted with limited human assistance.

[1] Due to limitations in PDDL, actions in the pipesworld are split into start and finish actions. Our reasoning questions here treat such pairs as single actions.

## Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 0905372.

## References

Bacchus, F., and Kabanza, F. 2000. Using temporal logics to express search control knowledge for planning. *Artificial Intelligence* 116.

Fox, M., and Long, D. 1998. The automatic inference of state invariants in TIM. *Journal of Artificial Intelligence Research* 9:1.

Hoffmann, J., and Edelkamp, S. 2005. The deterministic part of IPC-4: An overview. *Journal of Artificial Intelligence Research* 24:519–579.

McAllester, D., and Givan, R. 1993. Taxonomic syntax for first order inference. *Journal of the ACM* 40(2).

McAllester, D. 1989. *Ontic: A Knowledge Representation System for Mathematics.* MIT Press, Cambridge, MA.

McAllester, D. 1991. Observations on cognitive judgements. A.I. Memo No. 1340, Artificial Intelligence Laboratory, Massachusetts Institute Of Technology.

Srivastava, S.; Immerman, N.; and Zilberstein, S. 2010. A new representation and associated algorithms for generalized planning. *Artificial Intelligence* 175:2:615–647.