Online Hierarchical Conformance Refinement Planning for Intelligent Autonomous Mobile Robots using Answer Set Programming

Oliver Michael Kamperis (1332412)

School of Computer Science, University of Birmingham, United Kingdom

Supervisors: Marco Castellani and Yongjing Wang

Thesis Group: Dave Parker and Christopher Davis

Introduction

This thesis proposes a novel approach to planning called hierarchical conformance refinement. The objective is to make complex real-world discrete planning problems feasible to solve online, whilst requiring the robot to understand only the fundamental physical laws of the system in which it operates. The idea is to generate plans over an abstraction hierarchy, containing multiple models of the planning domain and problem, at different levels of abstraction. Plans are then progressively "refined" downwards over the hierarchy, under a constraint that requires the plans achieve the same effects and remain structurally similar at all levels. This provides the following major benefits:

Problem Simplification: Abstract models can reduce an exponential search space to linear, allowing abstract plans to be found exponentially faster than in the original model. The resulting abstract plan then guides search in the original model, by enforcing a conformance constraint formed by a sequence of sub-goal stages, defined by the effects of the abstract plan's actions. The original level plan must achieve these sub-goal stages in order. This restricts the search space whilst maintaining plan quality, and can reduce overall planning time exponentially, over directly solving the problem.

Problem Division: Abstract plans can be used to divide and sub-divide complete planning problems into a sequence of distinct partial problems, each defined by a contiguous subsequence of sub-goal stages. Only the first of these must be solved prior to execution, and can then be progressively extended during execution, towards eventual achievement of the final-goal. Under the assumptions that; minimal plan length scales linearly with problem size (in terms of number of sub-goal stages included), and planning problem complexity is exponential in the minimal plan length, then solving a sequence of partial problems reduces both; execution latency and total planning time, exponentially.

Motivations

Classical state space heuristic planners have been studied extensively. For many problems, such as path planning, where an accurate estimate of the cost or "distance" to the shallowest goal state can be found trivially, these planners are optimally efficient and extremely effective [1]. However, for more complex combined problems, such as those including locomotion and manipulation, it can be more difficult to form an accurate heuristic, because there is rarely a direct way to measure the distance to the goal [2]. Some methods use elaborate predictive models to provide these estimates based on knowledge learnt from prior experience, but these can be prohibitively expensive to use or obtain. Resultantly, classical planners are usually not very expressive, having to keep to a restricted problem representation to allow general reasoning about the laws of a system by the heuristic [3].

ASP based planning has seen significant interest on this front. Due to its foundations in logic programming and knowledge representation, it is known as one of the most highly expressive and general discrete deterministic planning paradigms available [4]. Planning problems are represented by encoding the physical laws of a system as logical axioms, e.g. “when you move, your location changes” or “a square peg can’t go in a round hole”, which are intuitively understood and defined by a human engineer. These axioms tell the planning agent not what to do or when to do it, but rather only what its capabilities are and what constraints its domain poses upon it. The satisfiability solver used by ASP is then able to reason for itself about how to solve a problem. ASP can perform well for challenging combined problems, because the solver can reason efficiently with complex physical system laws without specialised heuristics [5]. It is also highly scalable to large knowledge bases, making it effective for large problems which include many relevant entity constants. However, it is still inherently limited in its usability, because it rapidly becomes intractable for long plan lengths [6].

Refinement planning has on the other hand, proved effective at reducing planning complexity via a divide-and-conquer technique [7]. Instead of heuristics, refinement planners use an abstraction hierarchy, to divide and sub-divide planning problems into a sequence of sub-problems. This works simply by generating an abstract plan in a relaxed version of the planning problem (where enabling conditions of actions may be ignored) [8]. Each action of an abstract plan is then refined separately using its start state and enabling conditions, to form the initial and goal states respectively, of a sub-problem at the next (more concrete) level. The central concept is that for complex problems, these abstractions might be much easier to obtain than the equivalent heuristic. Indeed, the technique can be effective for problems with long minimal plan lengths, because the complete plan is essentially split into shorter sub-plans [9]. However, this did not always prove true for two reasons; (1) existing planners have had very limited capacity to express abstract models, and (2) there exists an implicit assumption of independence between the sub-problems used, which fails often [10]. Ultimately, these mean that you cannot always find an appropriate abstraction for most interesting planning domains and problems, nor is it necessarily possible to produce meaningful divisions of a problem.

A large motivator for refinement style planning, is the apparent necessity of the use of abstraction in reasoning in human planning and problem solving [11]. The belief was that by emulating this human ability, a highly general paradigm could be made, that would be more effective for complex real-world problems than heuristic planners. However, existing refinement planners are simply not nearly as capable or flexible at representing the necessary abstractions or refining plans as humans are.

Contributions

This thesis provides a combination of ASP based and hierarchical refinement planning, and finds in their complementary capabilities, the overcoming of their weaknesses and the reinforcement of their strengths. The resulting planning paradigm and planner implementation is expressive and general, with the ability to represent and plan with only the fundamental physical laws of a system, to solve large problems with long minimal plan lengths that would otherwise be intractable to solve in ASP. The proposed approach also alleviates the two major problems of past refinement planners.

1. A novel ASP encoding allows representation of an arbitrarily large abstraction hierarchy, supporting any type of abstract model to which an exhaustive and deterministic state space mapping can be defined from the original model to the abstract. This allows the use of any form of abstraction that might be affective for a given domain. If the state and action spaces are not changed by an abstract model, its abstract system laws are generated automatically, the designer must only specify which laws to simplify away (or possibly replace). If the state and action spaces are changed, the designer must also specify the state space mapping.

2. The sub-problems used in past refinement planners are generalised into partial-problems. Each action of an abstract plan produces a sub-goal stage (rather than a sub-problem), partial-problems then involve refining any given contiguous subsequence of sub-goal stages (equating to refining a subsequence of abstract actions) produced from an abstract level, rather than just a single abstract action. The only constraint applied is that these sub-goals must be achieved in the same order that the actions that produced them were planned. This is critical because it introduces the novel interleaving property, which allows all sub-goal stages included in the same partial-problem to be considered and pursued simultaneously. Essentially, this property allows the achievement of earlier sub-goal stages to be delayed, in order to prepare the state for conditions that are required to achieve later sub-goals sooner, if this would reduce the overall plan length. This relaxes the assumption that there is an independence between the refinement of abstract actions in the same partial-problem.

The planner performs well for the types of problem it was designed for. These are complex problems with many interacting constraints and action enabling conditions. It is intuitively best when there are clear abstractions that can be made (e.g. simplifying assumptions or removal of problem details) and where there exists natural divisions of a problem. These problems can be intractable to solve by ASP based classical planning, unless the plan lengths are trivially short. However, for problems where an accurate heuristic estimate is easily measurable, and an abstraction of a problem cannot be so easily made (most notably for example is pure path planning, where no aspect of the problem lends itself directly to abstraction or problem division), conformance refinement can perform very poorly.

The main challenge in conformance refinement planning, is the construction of the abstraction hierarchy itself, and the tailoring of it towards a specific domain/application. There are two key points here; (1) if the problem is sufficiently complex, constructing an abstraction hierarchy can in theory be much easier than forming a heuristic or predictive model to guide search, as constructing the hierarchy requires only removing descriptive knowledge, (2) spreading the task of planning over a hierarchy allows complexities of a problem to be broken down and dealt with in natural “phases” as the constraints/details of the original problem are progressively reintroduced down the hierarchy.

This thesis explores the following forms of abstract model in order to exemplify this technique:

1. **Relaxed Models:** Contributed and used heavily in past work, a well understood method to creating abstractions. These create a less constrained problem and allows the use of actions in states in which they would usually be prohibited. This makes abstract plan lengths shorter because less actions are necessary to achieve, or to enable those that achieve, the final-goal.
2. **Condensed models:** The main novel abstract model contributed by this thesis. These simplify state and action representations to reduce search spaces and plan lengths, by lowering the number of necessary actions required to reach the goal. This works simply by combining sets of related “constituent descendent” entities into abstract descriptors, which serve to represent the union of those constituents (and their properties) that compose them and together form some greater whole. For example, we might define a room to be an abstract description for the set of constituent discrete cells that make up that room. If we then abstract away those cells and reason only about the room in general, we reduce the state and action representations related to that room by a factor of the number of cells inside it.
3. **Tasking models:** Uses high level application-specific “task description actions” and bespoke specialised abstraction mappings which serve as conditions for completing those tasks. This provides a powerful method to decompose sub-sets of related literals in the final-goal that can be said to form part of such a distinct task embedded in the larger complete planning problem. Each of these tasks can then be dealt with as a separate partial planning problem.

Fundamental Assumptions in Conformance Refinement

The planner implementation operates under the following fundamental assumptions. These are limitations to the capabilities of the planner, but relaxing them is outside the scope of this thesis.

Completeness of Knowledge Assumption: The problem specification is assumed complete and consistent. The planner must know all the relevant system laws, the complete initial state, and the domain's static structure (this includes all entities involved), before planning and with certainty.

Existence of Solution Assumption: The problem is assumed to have a valid solution, such that there is some plan that transitions the initial state to a goal state. The planner only returns when a solution is found, and will never return if one does not exist. Formal proofs of solution existence are not the focus of this thesis, but a description of the requirements that would need to be met, and why it is so difficult to prove in a general way (i.e. given only the problem specification) will be described.

Determinism of Execution Assumption: Generated plans are assumed to never fail, because the plan's execution is deterministic, i.e. it will progress exactly according to the expected simulated sequence of states and transitions. Hence, although the proposed approach and provided planner can yield partial plans online, it does not deal with plan failure, diagnostics, or re-planning.

Fundamental Problems in Conformance Refinement

The following are fundamental design, reasoning, and decision-making problems, which have been found to occur in conformance refinement from the experiments that have been complete thus far.

The Ignorance Problem: Decisions made in an abstract version of a problem do not (by design) consider all knowledge and planning constraints of the original, or in other words they are ignorant of the full nature of the original problem. Although this makes the problem easier to solve, it may also make the quality of abstract plans worse. This occurs because there will commonly be multiple possible actions/plans at the abstract level that appear equally good quality according to the simpler constraints considered at that level. However, when refined under a conformance constraint at the original level, many of these plans might not be optimal under its more elaborate constraints. It is very difficult to avoid this problem, as it occurs naturally according to the technique. However, current results show that the reduced plan quality is far outweighed by the reduced planning time.

The Dependency Problem: Although conformance refinement generalises sub-problems into more flexible partial-problems, there is still an implicit assumption of independence between sub-goal stages that are in different partial-problems. There can thus still exist dependencies between partial-problems, although these are usually much weaker than would exist between sub-problems. The dependency problem can cause a much larger reduction in plan quality than the ignorance problem. This is because the achievement of sub-goal stages in earlier partial-problems is "greedy" with respect to those in later partial-problems. This greedy achievement occurs because the global outlook of the solution to the partial-problem is not considered, and the end state of the resultant partial-plan might make it harder to solve the next partial-problem. If the quality of plans is reduced enough, the speed of the planner might suffer too much for trade-off to be worthwhile. This hints at a fundamental decision-making challenge in problem division. The larger the partial-problems are, the higher their complexity, but the more sub-goals are allowed to interleave. Larger problems will thus increase planning time, but lower the chance of dependency and increase plan quality, and vice-versa for smaller problems. It is however, inherently difficult to predict the complexity of (previously unsolved) partial-problems or chance of dependency between them, prior to planning.

The Unconsidered Final-Goal Problem: Because sub-goals are produced from the effects of abstract actions, and thus they are defined by the abstract state space which may be less detailed than the original state space (as occurs with condensed models), achieving a sub-goal stage produced from an abstract action that achieved an abstract final-goal literal, might not achieve the corresponding original level final-goal literals(s). This is because there may be many possible original level state literals that map to the abstract level state literal in the sub-goal stage. As a result of this, partial-problems that don't include the final-goal (all except the last) can end in a state that is arbitrarily far from any final-goal achieving state, depending on the extent of the state space abstraction.

Proposed Methods to Overcome the Problems

Concurrent Action Planning: One of the most common causes of the ignorance problem, occurs when a subsequence of actions in an abstract plan are ordered arbitrarily. That is, if there are multiple different orderings of those actions which all result in a plan of equal quality, the abstract level planning must choose one of these orderings, essentially at random. However, usually only a sub-set of these possible orderings will refine to an optimal plan at the original level, yet at the time of abstract level planning, the planner is ignorant of which ones. To help alleviate this problem, this thesis provides a novel general method for dealing with action concurrency. This allows a set of actions to be planned on the same step at the abstract level, and when refined at the next level, the sub-goals produced from the effects of those actions can be achieved in any order, thus retaining the knowledge that there are no abstract constraints that require a strict order over those actions. Current results show that this can alleviate the ignorance problem, but (for reasons that are difficult to describe concisely), it can increase the dependencies between different sub-goal stages.

Problem Division Strategies: This thesis provides a variety of different novel problem division strategies for making divisions and sub-divisions in a flexible manner. These strategies decide how many sub-goal stages are included in each partial-problem. They range from; simple naïve proactive strategies, which make a sequence of homogenous (equal size) partial-problems according to a given size bound and a preference on favouring speed or plan quality, to varying reactive strategies that use a feedback function from the planner and can make divisions to the problem during solving based on a variety of search time bounds (e.g. incremental, integral, differential, and predictive) which is beneficial because it gives finer control over the speed and progression of the planner. More complex adaptive informed strategies are also possible, that estimate the problem complexity or chance of dependency to form an initial tentative proactive strategy and then reactively commit additional divisions as required. In previous refinement planners, there was no capacity to use such strategies, since there was no flexibility in the construction of the sub-problems they used.

Problem Blending: A conceptually very simple technique, that partly merges together adjacent partial-problems by a "small" number of neighbouring sub-goal stages. The idea is that this keeps the achievement of the sub-goal stages in the earlier partial problem non-greedy with respect to the sub-goal stages in the right-blend (the part of the later partial-problem merged with the earlier), and the achievement of the sub-goal stages in the left-blend (the part of the earlier partial-problem merged with the later) non-greedy with respect to the later partial-problem. Current results show that this can heavily alleviate the dependency problem, but only between adjacent partial-problems.

Final-Goal Pre-emptive Achievement Heuristics: Another simple technique, that uses a general heuristic to bias search towards choosing actions during solving earlier partial problems, which achieve final-goal literals, if the resulting plan is both still valid and is not reduced in plan quality. Current results show that this can completely eliminate the unconsidered final-goal problem.

Summary of Progress and Remaining work

I have decided on the content of the thesis, the way I am going to pose the contributions in the context of existing work, and have written down and formalised most of the theory now. I am now also close to completing the planner implementation, only some unit testing and documentation remains to be finished. This planner will be released as free-ware supporting the thesis.

Following this, the bulk of the remaining work involves producing additional experimental domains, and the experimental testing of these domains under a variety of different planner configurations. I expect to be able to complete these experiments before the end of my research period (this year).

My write up period is the entire of 2022. I have already written approximately a third of the thesis in the form of a paper, much of the rest of the thesis has been written in rough, but most of the analysis of experimental results (bar those obtained in initial experiments form early this year) is still to be done. I am producing a detailed plan of the thesis chapters; a condensed version follows.

Plan of thesis chapters

The thesis chapters and major section headings are listed below.

1. **Introduction:** A concise overview of the thesis contributions, the concepts of the proposed approach, the capabilities of the planner, and the types of planning problems focused on.
    a. *Contributions and Proposed Approach*
    b. *Fundamental Assumptions*
    c. *Overview of Thesis Structure*
2. **Literature Review:** Places the thesis and its contributions in the context of existing work, and motivates the proposed approach according to the weaknesses of that existing work that it seeks to overcome for the problems focused on. This chapter will cover a number of topics, many topics will be obscure to most readers, so will likely require examples for clarity.
    a. *Why Intelligent Autonomous Robots need to Plan*
    b. *Classical Planning*
    c. *ASP based Planning*
    d. *How Humans Plan: Abstraction in Reasoning*
    e. *Abstraction and Hierarchies for Automated Planning*
3. **Fundamentals of Conformance Refinement Planning:** The bulk of the theory and design ideas of the proposed approach and planner will be described in this chapter. This includes, definitions of planning domains, problems, and the algorithms used to find their solutions. Initial experiments will be done, to motivate and support the benefits and trade-offs (when and how conformance refinement is effective), exemplify the main concepts, and identify the fundamental problems.
    a. *The Blocks World Plus Example Domain*
    b. *Planning Domain Definitions and Domain Models*
    c. *Creating a Condensed Abstract Domain Model*
    d. *Planning Problems, Solutions and Algorithms*
    e. *Design of ASH: The ASP based Hierarchical Conformance Refinement Planner*
    f. *Initial Experiments*
4. **Optimisations and Enhancements:** Methods for overcoming or alleviating each of the fundamental problems are presented in this chapter. This will show that each problem has been addressed at least partially, and the challenges in doing so are understood and exemplified clearly. This is not an exhaustive study of all the conceivable methods that could be used, but instead identifies the general properties/concepts that may be exploited to improve the planner.

    a. *Concurrent Action Planning*
    b. *Division Strategies and Scenarios*
    c. *Dependency and Complexity Prediction*
    d. *Partial Problem Blending*
    e. *Final-goal Pre-emptive Achievement*
    f. *Tasking Models*
5. **Experimental Studies:** This thesis will take a pragmatic approach to justify and demonstrate the strengths of the proposed approach. This chapter will provide extensive experimental testing of three different planning domains and varying hierarchy structures for each. Different division strategies and planner configurations will be explored for each of these, and the results used to identify which and when each performs effectively, and where there is room for improvement.
    a. *Experimental Domains and Setups*
    b. *Experimental Results*
    c. *Evaluation and Discussion*
6. **Conclusions:** Concisely reiterate the contributions of the thesis, using results of the experimental studies to support and justify conclusions drawn and conjectures made. Identify the areas for continued future work including; more advanced problem division strategies, different abstract models and hierarchy structures, and relaxation of the fundamental assumptions of this thesis.
    a. *Summary of Contributions and Findings*
    b. *Future Work*

References

[1] Hart, P.E., Nilsson, N.J., Raphael, B., 1968. A formal basis for the heuristic determination of minimum cost paths. IEEE transactions on Systems Science and Cybernetics 4, 100–107.

[2] Helmert, M., 2003. Complexity results for standard benchmark do-mains in planning. Artificial Intelligence 143, 219–262.

[3] Knoblock, C.A., 1990b. A theory of abstraction for hierarchical planning, in: Change of Representation and Inductive Bias. Springer, pp.81–104.

[4] Gelfond, M., Kahl, Y., 2014. Knowledge Representation, Reasoning, and the Design of Intelligent Agents: The Answer-Set Programming Approach. Cambridge University Press, New York, NY, USA.

[5] Gebser, M., Kaminski, R., Kaufmann, B., Schaub, T., 2019b. Multi-shot asp solving with Clingo. TPLP 19, 27–82.

[6] Jiang, Y.q., Zhang, S.q., Khandelwal, P., Stone, P., 2019. Task planning in robotics: an empirical comparison of pddl-and asp-based systems. Frontiers of Information Technology & Electronic Engineering20, 363–373.

[7] Sacerdoti, E.D., 1974. Planning in a hierarchy of abstraction spaces. Artificial intelligence 5, 115–135.

[8] Knoblock, C. A., Tenenberg, J. D., and Yang, Q. (1991). Characterizing abstraction hierarchies for planning. In AAAI, pages 692–697.

[9] Knoblock, C. A. (1990). A theory of abstraction for hierarchical planning. In Change of Representation and Inductive Bias, pages 81–104. Springer.

[10] Bacchus, F., Yang, Q., 1994. Downward refinement and the efficiency of hierarchical problem solving. Artificial Intelligence 71, 43–100.

[11] Newell, A., Simon, H. A., et al. (1972).Human problem solving, volume 104. Prentice-Hall Englewood Cliffs, NJ.