Centralised Online Hierarchical Abstraction Planning and Diagnostics for Heterogeneous Robot Teams

PhD Thesis

Oliver Michael Kamperis, Marco Castellani and Yongjing Wang
School of Computer Science
University of Birmingham
United Kingdom

December 8, 2020

1 Progress Report

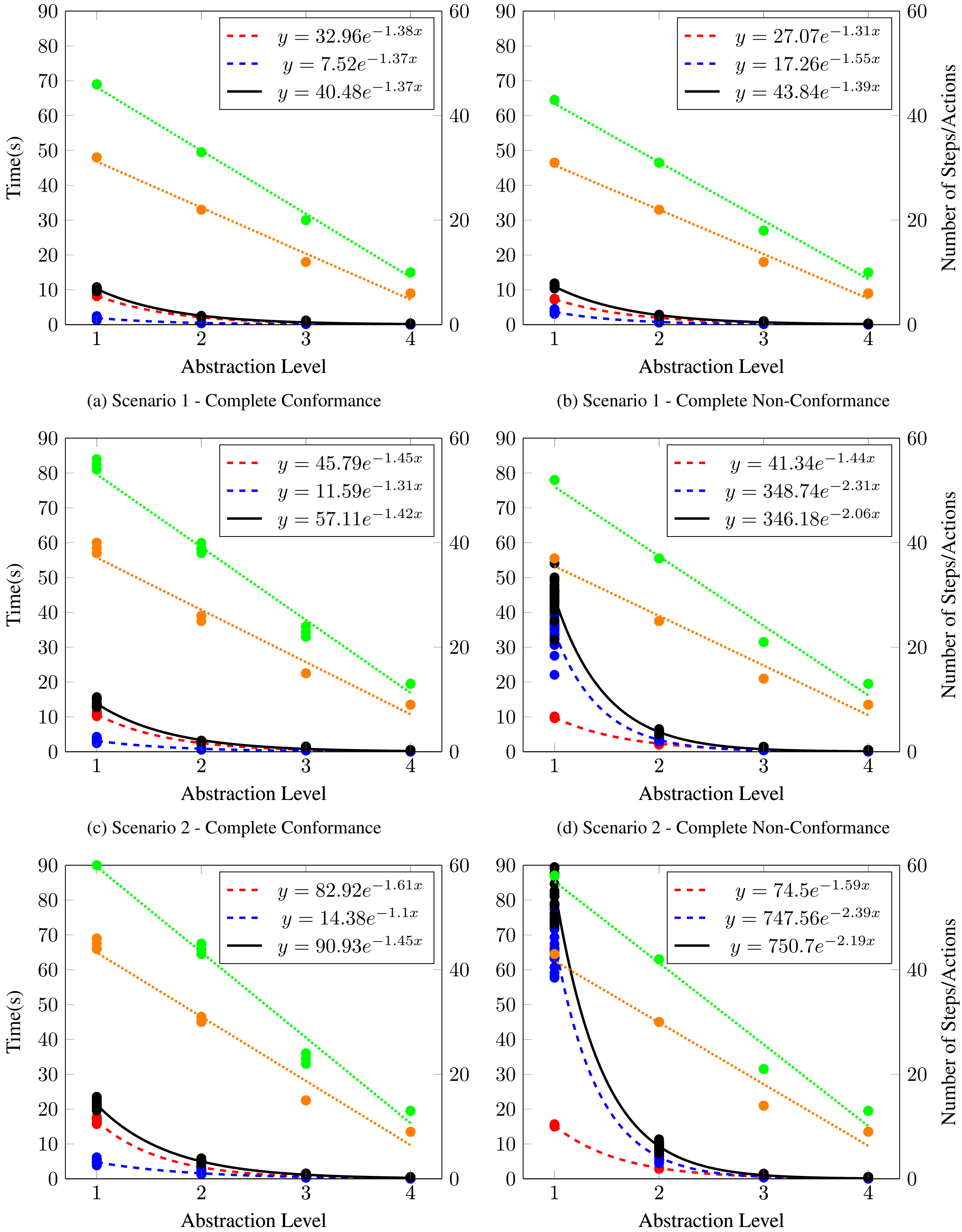
Since our last thesis group meeting I have been working towards three primary objectives; (1) solidifying my theories and proposed approach to hierarchical abstraction planning, (2) implementing a software instance of the proposed approach in Answer Set Programming (ASP), (3) and experimentally validating that implementation in simulation. There has only been two significant marked changes in the proposed work, both made as suggested by the thesis group; (1) The focus on Human-Robot collaboration has been removed and shifted towards online planning, (2) the construction planning domain has also been removed so that I may focus on disassembly sequence planning.

The proposed approach is founded upon the use of abstraction in planning and reasoning, which has two important benefits [Knoblock 1990]; (1) *problem simplification* - abstraction can reduce an exponential search space to linear, allowing solutions to be found exponentially faster, with the resulting abstract solution guiding the search for a solution in the concrete space, (2) *problem splitting* - abstract solutions can be used to infer distinct sub-problems in the concrete space which can be solved independently. The current software implementation is called ASH, the Answer Set programming based Hierarchical Abstraction planner. ASH generates plans over an arbitrary number of abstraction levels (layers or spaces) organised hierarchically. Plans are generated at every abstraction level in the hierarchy in descending order. The plan from each abstraction level is used to provide both of the above benefits when planning at the next level, granting heuristic guidance to search and allowing problem splitting which enables partial planning.

The proposed approach is intended to make even very complex discrete deterministic planning problems possible to solve online. We have taken some relatively limited experimental results which indicate this is achievable/has been achieved, but it has been very difficult to collect sufficient and "clean" experimental results with only access to my personal computer. Ideally, we intend to test the current implementation of a much larger array of problems of various complexity, including problems of extreme complexity, using the university's Bluebear supercomputer. This would give a clearer picture of the effectiveness of the proposed approach and make our conclusions much more convincing.

1.1 New Results

Some of our experimental results are given in Figure 1. It displays plots of plan quality and computation time over all abstraction levels for conformance planning (left-hand-side) and non-conformance complete planning (right-hand-side) over all abstraction levels for three different planning scenarios of increasing complexity in ascending order from top to bottom. Where conformance planning is effectively planning using the hierarchical structure to provide heuristic search guidance, and non-conformance planning simply does not use the structure. The step and action values are measures of plan quality, where lower values indicate better quality plans that requires less time and work to execute. The grounding, solving and total times are measures of the cost of generating the respective plan, where lower values indicate the solution was obtained faster. The grounding time is the time taken to obtain the ground logic program by replacing all non-ground logic rules with their ground instances. The solving time is the time taken to exhaust the entire search space and find the optional solution. In all instances we observe an inverse linear trend between step/action values and abstraction level, and we observe an exponential decay in computation time as abstraction level increases. Importantly, in all instances, we observe a significant reduction in solving time (and therefore total time) when using conformance planning over non-conformance planning, with the reduction becoming more distinct for more complex scenarios.



(a) Scenario 1 - Complete Conformance

(b) Scenario 1 - Complete Non-Conformance

(c) Scenario 2 - Complete Conformance

(d) Scenario 2 - Complete Non-Conformance

(e) Scenario 3 - Complete Conformance

(f) Scenario 3 - Complete Non-Conformance

Figure 1: The BWP domain complete planning results for all three benchmark scenarios.
Red, blue and black indicate grounding, solving and total time (s) respectively.
Orange and green indicate total steps and actions respectively.

Our current results indicate that the proposed approach performs better than past approaches, although we have not directly compared ASH's performance with other similar planners. We can compare the proposed approach to existing ASP based planners and other symbolic planners that do not use abstraction using existing studies [Jiang et al. 2019]. However, although there is a significant quantity of proposed hierarchical and abstraction based approaches to planning, many of these either; differ too greatly in their functional methods to ASH, or there are no available solvers for the approach in question. Existing literature for hierarchical planning is dominated by Hierarchical Task Network (HTN) planning and its extensions [Erol et al. 1994]. There is an array of available efficient HTN solvers as the approach is popular, however it is similar only in an abstract conceptual sense to ASH. Abstraction based planners that are similar to ASH in functionality, such as ABSTRIPS [Sacerdoti 1974], are on the other hand, to the best of my knowledge, not publicly available. This appears to be due to the apparent complexity of past approaches to abstraction planning.

1.2 Theoretical Limitations

We have encountered one major theoretical problem which has made it difficult to get good results for partial planning. ASH constructs hierarchical plans consisting of sequences of decomposition trees. A decomposition tree is a tree-like data structure that represents how the effects of abstract actions are achieved by sequences of more concrete actions, i.e. mapping plans between different abstractions. However, each decomposition tree in a plan may have very different complexity, and we do not have a precise way to measure their complexity. This is complicated because the perceived complexity of the fully constructed tree does not always appear to be indicative of the actual complexity of solving it.

Figure 2 is an abstract representation of a hierarchical plan containing unbalanced decomposition trees. The nodes are of the form $abstraction\ level : time\ step$, in an actual plan these would be sets of actions. The red dashed lines indicate places where the hierarchical plan could be split into individual decomposition trees. The middle-level trees whose heads are $2:1$ and $2:2$ have an unbalanced number of child nodes (i.e. descendent action sets). Despite this, it does not necessarily mean that solving $2:1$ takes longer than solving $2:2$ since this is dependent on factors not represented in the tree. Since partial planning essentially involves constructing a sub-set of the trees that make up a hierarchical plan, the complexity of each tree becomes the most import factor informing the decision of how many to solve. We do not currently have a solution for this or a way to create planning problems were all the trees are balanced.

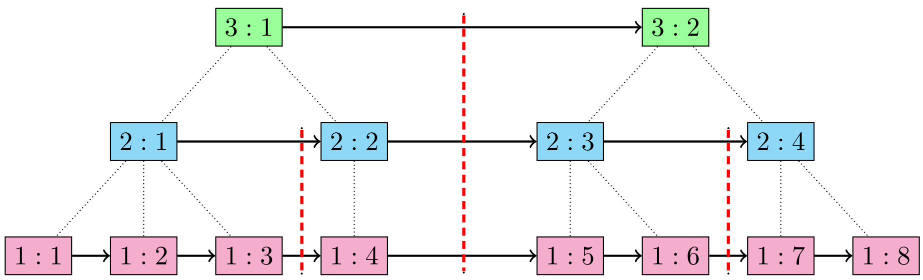


Figure 2: An abstract Hierarchical Plan containing Unbalanced Decomposition Trees.

2 Future Work

The following are my current three working thesis "pyramids", each of which we hope to publish as a separate paper. Those papers will then be used to form the foundations of the final thesis during the write-up stage of the thesis.

1. **Centralised hierarchical abstraction planning and online partial planning using ASP for heterogeneous robot teams:** This has been the focus of our work since the last thesis group meeting. At the current time, the implementation of ASH fully supports this pyramid but we have not yet experimented with multiple robots.
2. **Centralised diagnostics of unexpected action/goal/plan failure and hierarchical plan repair:** This will be the focus of our work once we are happy that the results obtained for the prior pyramid are sufficient. Essentially, the objective of this pyramid is for ASH to interface with real robots, receiving observations returned from the robots during execution. When an unexpected observation is received that invalidates a plan, ASH will resolve the contradiction between belief and observation and repair the existing plan. If any goal is made unachieveable by this observation, ASH will attempt to achieve the highest utility sub-set of goals.
3. **Case study for battery electric vehicle lithium-ion battery disassembly sequence planning:** This is a challenging problem which is receiving a lot of attention in research and the media due to its economical and environmental impacts. It is a problem to which ASP based approaches have not before been applied to and it also presents all the problems that ASH is designed to overcome, making is an excellent case study choice.

2.1 Thesis Extension

At the current stage in the thesis, it seems very likely that we will need to request an extension. The COVID-19 pandemic, the change of supervisors and the shift in thesis topic have all caused significant delays and setbacks. The deadline for requesting an extension to the deadline based on a COVID-19 is 1st March 2021. We intend to wait until closer to this date to apply to determine the full extend of the disruption COVID-19 has caused to my progress.

The pandemic has been particularly difficult for me. Marco and Yongjing have been very helpful and supportive, most weeks we have spent up to an hour talking via Skype/Zoom, but there is only so much one can do and convey over the internet. I was put in the rather unique situation that my supervisor change occured immediately before the lockdown started which has complicated matters because it has very difficult for me to provide a clear picture of my work without meeting face-to-face. Further, since I also found the supervisor change very stressful and felt that I must to distance myself from my old topic, much of my work from the first half of the second year is now irrelevant.

2.2 Plan

The current plan of future work up until the end of year 3 of this thesis is shown in Figure 3. The plan is a worst case scenario assuming that no extensions are granted. It includes some optional tasks which may be dropped in such case.

At present, a significant amount of work has already been done towards implementing ASH, and performing initial experimental testing. Starting in the first week of November 2020, following the RSMG 5, we will thus be working primarily towards publishing this work. We also intend to publish ASH's code freeware on Github to accompany the paper, so some time will be spent documenting the code and providing usage instructions via online appendix. ASH is intended to be usable on real robots, so it is important that we include fully functional software with the publication.

In early 2021 we will need to review what is feasible to achieve with the time available and submit for an extension if necessary. If there is spare time, we will preform additional experimental testing of the current version of ASH against other similar planning paradigms via Bluebear. During this time, we will also implement the diagnostic and plan-repair systems into ASH for the second thesis pyramid. This will be presented in the RSMG 6 meeting in April 2021.

Following the RSMG 6 we will then work towards publishing these contributions from the second thesis pyramid. We will also devote significant time towards developing an ASH compatible model for disassembly sequence planning problems for the third thesis pyramid during this period. It is however not yet clear what the nature and complexity of the disassembly problem we will tackle will be and whether experiments will occur on real-world robots or in simulation. We hope to have sufficient time to evaluate the effectiveness of our approach against previous approaches to disassembly planning which used existing methods such as the Bees algorithm [Laili et al. 2019, Xu et al. 2020]. This work will be written up into a third publishable paper and presented in the RSMG 7 meeting in November 2021.

If an extension is granted, we will do the following. A 3 month extension would allow us to focus on getting better experimental results via Bluebear and compare the proposed approach to other planning paradigms. We will add an additional month to each set of experimental testing periods (shown in green) in this case. With a 6 month extension will we devote the extra time towards performing disassembly sequence planning experiments on real robots in 2022.

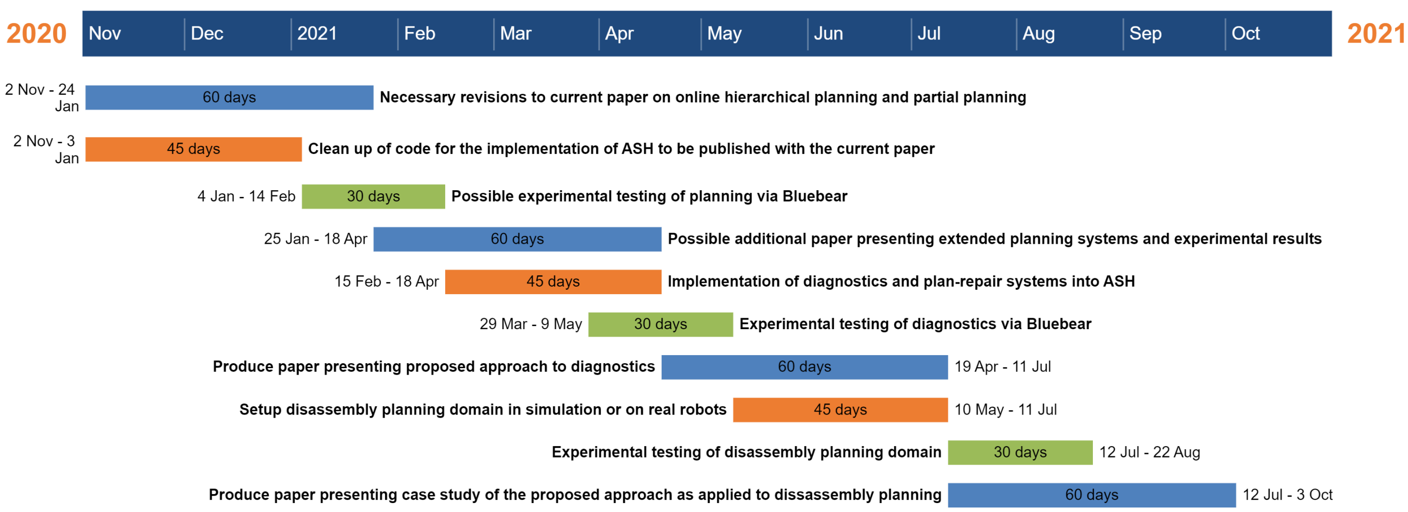


Figure 3: Plan of future work up until the end of year 3.

References

[Erol et al. 1994] Erol, K., Hendler, J., and Nau, D. S. (1994). Htn planning: Complexity and expressivity. In *AAAI*, volume 94, pages 1123–1128.

[Jiang et al. 2019] Jiang, Y.-q., Zhang, S.-q., Khandelwal, P., and Stone, P. (2019). Task planning in robotics: an empirical comparison of pddl-and asp-based systems. *Frontiers of Information Technology & Electronic Engineering*, 20(3):363–373.

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

[Laili et al. 2019] Laili, Y., Tao, F., Pham, D. T., Wang, Y., and Zhang, L. (2019). Robotic disassembly re-planning using a two-pointer detection strategy and a super-fast bees algorithm. *Robotics and Computer-Integrated Manufacturing*, 59:130–142.

[Sacerdoti 1974] Sacerdoti, E. D. (1974). Planning in a hierarchy of abstraction spaces. *Artificial intelligence*, 5(2):115–135.

[Xu et al. 2020] Xu, W., Tang, Q., Liu, J., Liu, Z., Zhou, Z., and Pham, D. T. (2020). Disassembly sequence planning using discrete bees algorithm for human-robot collaboration in remanufacturing. *Robotics and Computer-Integrated Manufacturing*, 62:101860.