Planning with Concurrent Execution

Bence Cserna
Department of Computer Science
University of New Hampshire
Durham, NH 03824 USA
bence@cs.unh.edu

Abstract

In real world planning applications such as planning for robots or video games, time for decision making is often limited. Unlike classical search, real-time search algorithms are applicable in these domains due to their bounded response time. This dissertation addresses the setting in which planning and execution occur concurrently using real-time heuristic search. I advance this area in three ways.

Real-time planning algorithms have to return the next action for the agent within a strict time bound. I show how to adapt previously-proposed methods, which depend on unrealistic assumptions, to allow the use of wall clock time bounds. Second, in real-time search partial plans are generated until the goal state is found. It is not clear whether the agent should commit to one action along the path to best explored node or all the way to the search frontier. We investigate the use of metareasoning algorithms to decide how many actions to commit to. Third, I will extend the application of real-time search algorithms to stochastic domains.

Introduction

In real-time planning domains where human interaction is involved, it is often desirable to reach the goal state as fast as possible. Consider a path-finding domain in which an agent has to navigate to a goal location while the user waits for the execution to be completed.

The performance of classical heuristic search algorithms is often measured by the solution cost, the overall search time and the expansion count. These classical heuristic search metrics are not sufficient for such real-time domains. In problems where the agent has limited time to act the performance of the planning algorithm is measured in goal achievement time (GAT). The GAT measures the wall time from the start of the search to the completion of the task.

Our primary focus is to advance real-time search in three different ways.

Real-time search algorithms operate in planning episodes. After each episode, the planner returns an action or a set of actions until the goal state is expanded. To satisfy the real-time property, the time to complete an episode is bounded.

The current state-of-the-art real-time search, Dynamic $\hat{f}$ provides responsiveness by limiting the number of explored states in the planning episodes(Kiesel, Burns, and Ruml 2015). The time bound of the algorithm is expressed in node expansions. According to the best of the author's knowledge none of the real-time heuristic search algorithms utilize wall time-bound nor are they feasible for such setup. However, in practical applications, the time bound is expressed in wall time, not in node expansion. We propose to use wall time to bound the response time of real-time planners.

In each episode, the planner explores the state space knowing the agent's current location and decides how far it should go in the explored space. Given the framework where the agent acts in parallel to the planning, the time the agent spends on executing actions establishes the time bound for the next planning iteration. The time limit given to the planner determines how far it can look ahead in the search space. More time allows the planner to explore more states and make a more informed decision. It is not clear how many actions should the agent commit to along the path to the best explored state on the search frontier. The early real-time search algorithms only committed to one action, but the leading algorithm, Dynamic $\hat{f}$ sends the agent all the way to the search frontier. We investigate the use of metareasoning algorithms to dynamically decide how many actions to commit to.

Additionally, unlike in classical benchmarks problems, acting in the real world is often stochastic. Consider an environment where a robot has to navigate to a goal location. The location of the agent after taking an action is non-deterministic. The precise location is unknown the planner can only estimate. The leading real-time search algorithms do not consider uncertainty, thus are not applicable to such domains. We propose to extend the existing real-time planning algorithms to domains with uncertainty.

Time bounded real-time search

The first real-time heuristic search algorithms were proposed by Korf in 1990. Since then the area of real-time search has been evolved and several improvements have been proposed. Before discussing the improvements, first we review the most relevant algorithms to our work.

Previous work

A significant drawback of using A* on a situated agent is that the algorithm has to find a complete solution path before making a commitment to the first action. The optimal first move cannot be guaranteed until a complete solution is found.

Learning Real-time A* (LRTA*) is a suboptimal real-time search algorithm (Korf 1990). Unlike A*, LRTA* only considers a local search problem. To select an action LRTA* chooses the successor state that has the lowest sum of cost and cost-to-goal estimate. It commits to one action and repeats the search from the selected successor state. The heuristic value of the neighboring states is augmented by a depth bounded lookahead. The action selection time has a theoretical upper bound due to the depth bounded lookahead.

Local Search Space - LRTA* (LSS-LRTA*) is an improved version of LRTA* (Koenig and Sun 2008).

A search episode of LSS-LRTA* consists of two phases: exploration and learning. In the exploration phase, the algorithm expands the search tree from the agent's current location. This exploration is limited by a predefined expansion bound. The states explored in each episode represent the local search space. While LRTA* selects the best neighboring state and commits to one action per search episode, LSS-LRTA* moves the agent all the way to the search frontier.

After the expansion, LSS-LRTA* uses a learning phase to propagate back the heuristic values from the search frontier in order to help the agent escape the local minima of heuristic values.

Dynamic $\hat{f}$ is a variant of LSS-LRTA* with two improvements. Instead of using and admissible heuristic to explore the local search space, Dynamic $\hat{f}$ utilizes an unbiased inadmissible heuristic function (Kiesel, Burns, and Ruml 2015). In addition, Kiesel, Burns, and Ruml proposed a technique to translate time duration to expansion counts to make the algorithm feasible for practical domains with real-time requirements. Dynamic $\hat{f}$ uses a lookup table based on the available time for the upcoming planning episode. The table is populated by offline learning and contains a predefined set of expansion limits along with the corresponding measured mean planning time.

Real-time search using wall clock bounds

In domains where the time for decision making is limited, the responsiveness of the planning algorithm is essential. We assume the planner algorithm is running on an operating system that does not provide real-time guarantees, thus the running time of an expansion cannot be approximated precisely.

On such systems the state expansion times could be inconsistent, therefore the duration of an episode of an expansion count limited real-time search algorithm is unpredictable, while the bounded response time is the primary requirement.

We propose a real-time variant of LSS-LRTA* that uses real-time measurements to provide more consistent planning episode duration. This could be achieved by replacing the expansion limit with a time limit. The proposed real-time algorithm assesses the time bound after every expansion.

An episode of LSS-LRTA* consists of two phases: exploration and learning. LSS-LRTA* does not account for the time of the learning phase, thus it is not limited by the expansion bound. Dynamic $\hat{f}$ is more sophisticated. It includes both the planning and the execution phases in the time bound by measuring the running time of the episodes. However, these estimates are calculated offline, therefore, they are not adaptive to the changes in the execution environment.

During the exploration phase of LSS-LRTA* or Dynamic $\hat{f}$, the planner expands the explored state space with new states until the time or expansion bound is reached. In the learning phase, the planner propagates back the heuristic values of the search frontier to local search space that were explored in the preceding exploration phase until every state is updated or the bound is reached.

The exploration phase is naturally ended by reaching the bound. However, the learning step is considered incomplete if only a subset of states were updated in the local search space. During the learning phase, the planner increases the heuristic values of the states in the local search space by propagating back the most relevant values from the search frontier. When the time bound is reached during the learning phase the propagation is terminated, leaving the planner in an inconsistent state. The partially updated local search space could be misleading due to the inconsistencies in the heuristic values. The planner would be biased towards the areas that were not updated, therefore, the results of an incomplete learning step should not be used.

The original LSS-LRTA* algorithm does not include the cost of learning in the episode bound due to the fact that it is faster to complete the learning phase than a corresponding exploration phase.

We propose to reverse the order of the learning and exploration in the episodes. The reverse order provides a better chance for the learning phase to complete. After updating the values of the local search space, the exploration phase can utilize the remaining time for the episode. The exploration time and the following learning phase duration are related. More exploration corresponds to longer learning time. When the learning step consumes most of the time in the episode the exploration step has less time to expand new states, thus, the learning step in the following episode will complete faster and leave more time for the exploration.

The proposed method makes LSS-LRTA* and Dynamic $\hat{f}$ feasible for practical real-time domains where the time bound is given in wall time.

A flexible commitment strategy

One of the central issues of real-time heuristic search with a situated agent is the relation between the agent and the plan. How far should the planner look ahead before committing to an action? Should it commit to only one action like LRTA* or should it be less conservative and commit to the best known state on the search frontier (LSS-LRTA*, Dynamic $\hat{f}$).

O'Ceallaigh and Ruml presented the first real-time search algorithm, Mo'RTS, that incorporates metareasoning to decide when to act, and how many actions to commit to (O'Ceallaigh and Ruml 2015). They showed the Mo'RTS can improve the GAT of leading real-time search algorithms, LSS-LRTA* and Dynamic $\hat{f}$. However, there are flaws that were acknowledged by the authors. First, the Mo'RTS is bounded by node expansions instead of using wall time bounds as LSS-LRTA*. The decision-making time was not included in the metric, thus, the effect of the algorithm on the GAT is not clear.

More importantly, the work of O'Ceallaigh and Ruml, while taking a principled approach to the question of whether or not the agent should commit to an action or deliberate further, takes an ad hoc approach to the question of how many actions the agent should commit to. I will show how, in fact, the second issue can be answer by appealing to the first. By showing that the metareasoning overhead of Mo'RTS can be made small in practice, I will allow the agent to consider the question of whether or not to commit very frequently — at multiple points during a single planning iteration, in fact. This will allow the agent to naturally decide how much of the plan prefix to which to commit, solving the commitment length problem without recourse to an ad hoc method.

The application of meta-reasoning to problems such as plan commitment has shown great promise but has not been fully explored. This dissertation aims to utilize such metareasoning techniques while relaxing the strong assumptions of Mo'RTS.

Extending to stochastic domains

A* based search algorithms are tailored to handle domains where the actions are deterministic. Thus, real-time search algorithms such as LSS-LRTA* and Dynamic $\hat{f}$ are not designed to handle stochastic domains, as plans generated by these algorithms assume certain state transitions.

Markov Decision Processes (MDPs) provide a popular framework for planning problems with uncertainty. Real-time dynamic programming (RTDP) is a generalization of Korf's LRTA* to stochastic domains (Barto, Bradtke, and Singh 1995). Unlike Dynamic $\hat{f}$, that explores only the local search space, RTDP explores states with simulated sequences from the agent's initial location. However, just as with traditional real-time, there has been little consideration of explicitly optimizing GAT. We intend to extend previous work, such as Sanner's Bayesian RTDP (Sanner et al. 2009), using ideas from Mo'RTS to show that MDPs can provide an agile real-time framework for situated agents planning under time pressure.

Conclusion

The primary focus of this dissertation is to overcome the existing issues of real-time heuristic search and to make real-time search actually real-time, while minimizing the goal achievement time. Furthermore, it aims to extend the scope of real-time heuristic search to handle a wider range of real world applications. My dissertation will advance the science of planning with concurrent execution in three ways. First, the current real-time search algorithms are not suited for planning in a real world environment in which the time for decision making is limited in wall time, since they provide responsiveness by limiting the number of expansions. Second, in a setting where the planning is concurrent with the execution, it is not clear whether the agent should commit to one action along the path to best explored node or all the way to the search frontier. Mo'RTS proposed an adaptive metareasoning technique to decide whether the agent should deliberate further, and I will adapt this method to naturally handle the length of commitment. Third, I will extend these ideas to planning under uncertainty.

References

Barto, A. G.; Bradtke, S. J.; and Singh, S. P. 1995. Learning to act using real-time dynamic programming. *Artif. Intell.* 72(1-2):81–138.

Kiesel, S.; Burns, E.; and Ruml, W. 2015. Achieving goals quickly using real-time search: Experimental results in video games. *Journal of Artificial Intelligence Research* 123–158.

Koenig, S., and Sun, X. 2008. Comparing real-time and incremental heuristic search for real-time situated agents. *Autonomous Agents and Multi-Agent Systems* 18(3):313–341.

Korf, R. E. 1990. Real-time heuristic search. *Artificial Intelligence* 42(2-3):189–211.

O'Ceallaigh, D., and Ruml, W. 2015. Metareasoning in real-time heuristic search. In *Eighth Annual Symposium on Combinatorial Search*.

Sanner, S.; Goetschalckx, R.; Driessens, K.; and Shani, G. 2009. Bayesian real-time dynamic programming. In Boutilier, C., ed., *{IJCAI} 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009*, 1784–1789.