Proceedings of the Ninth International
Symposium on Combinatorial Search (SoCS 2016)

Anytime versus Real-Time Heuristic Search for On-Line Planning

Bence Cserna, Michael Bogochow, Stephen Chambers,
Michaela Tremblay, Sammie Katt, and **Wheeler Ruml**
Department of Computer Science
University of New Hampshire
Durham, NH 03824 USA
bence at cs.unh.edu, {mgp36, mos25, smx227, sk1059} at wildcats.unh.edu, uml at cs.unh.edu

Introduction

Many AI systems, such as robots, must plan under time constraints. The most popular search approach in robotics so far is anytime search (Likhachev and Ferguson 2009), in which the algorithm quickly finds a suboptimal plan, and then continues to find better plans as time passes, until eventually converging on an optimal plan. However, the time until the first plan is returned is not controllable, so such methods inherently involve idling the system's operation before 'real' execution can begin. Real-time search methods provide hard real-time bounds on action selection time, yet to our knowledge, they have not yet been demonstrated for robotic systems. In this work, we compare anytime and real-time heuristic search methods in their ability to allow agents to achieve goals quickly. Our results suggest that real-time search is more broadly applicable and often achieves goals faster than anytime search, while anytime search finds shorter plans and does not suffer from dead-ends.

Our central performance metric is goal achievement time (GAT), which measures the time starting from when the agent receives a goal specification and begins planning to achieve it and ending when the agent reaches a goal state and is done with the task (Kiesel, Burns, and Ruml 2015). This metric provides a natural way to compare anytime algorithms, which form a complete plan before beginning execution, with on-line algorithms like real-time search.

One of the most well-known anytime search algorithms is Anytime Repairing A* (Likhachev, Gordon, and Thrun 2004) which runs a series of weighted A* searches. As soon as the first plan is found, the agent starts moving. However, in order for the previous search effort to remain applicable as the search continues for better plans, the search is actually performed starting at the goal state and searching towards the agent. This means that, even though the agent moves, the g-values (cost to a node from the original goal state) are still useful in the subsequent searches.

There are three important limitations of anytime search for concurrent planning and acting. First, searching backward from the goal requires the domain to have a predecessor function, which may be non-trivial. In contrast, real-time search can be used with any domain in which a forward simulator is available. Second, the agent cannot start moving until the first complete plan is formulated. In some systems, a delay in starting execution is merely undesirable, but in others such as fixed-wing aircraft, it is inherently infeasible. Third, because the time it will take for the search to find a new plan is not known in advance, it is not obvious what the goal state for the backwards search should be. In practice, a state along the current plan that is some predefined distance ahead of the agent can be selected, but this is ad hoc.

Real-time search algorithms, such as LSS-LRTA* (Koenig and Sun 2009), expand the search space until a hard real-time bound is reached, at which point the agent will commit to one or more steps towards the edge of the expanded search space. The heuristic values of expanded states are then updated by propagating information backward from the lookahead frontier. While the selected actions are being executed, the search computes the next action to take. In this way, the agent iteratively constructs a trajectory.

Real-time search suffers from two important limitations. First, it is incomplete in any domain in which there are dead end states from where the goal is unreachable. For example, the agent must be able to look far enough ahead to detect a fatal collision. Second, errors ('local minima') in the heuristic function can cause real-time search algorithms to visit the same states many times ('scrub'), in order to update the heuristic values inside the minimum. If the lookahead is smaller than the size of the minimum, repeated scrubbing can be required (Sturtevant and Bulitko 2014). This behavior appears irrational to an outside observer and is undesirable in many applications, including robotics (Likhachev 2016).

Experimental Results

We compared the performance of anytime and real-time heuristic search algorithms implemented in Kotlin using the IBM J9 Virtual Machine in several simulated domains: four-way grid pathfinding, grid pathfinding with inertia (Barto, Bradtke, and Singh 1995, 'racetrack'), a point robot in continuous space, a point robot with inertia ('double integrator'), and an underactuated double pendulum (Murray and Hauser 1991, 'Acrobot'). Adherence to time bounds was strictly checked. Heuristics were simple, such as Manhattan distance or Euclidean distance divided by the maximum velocity. Some of the results can be seen at https:

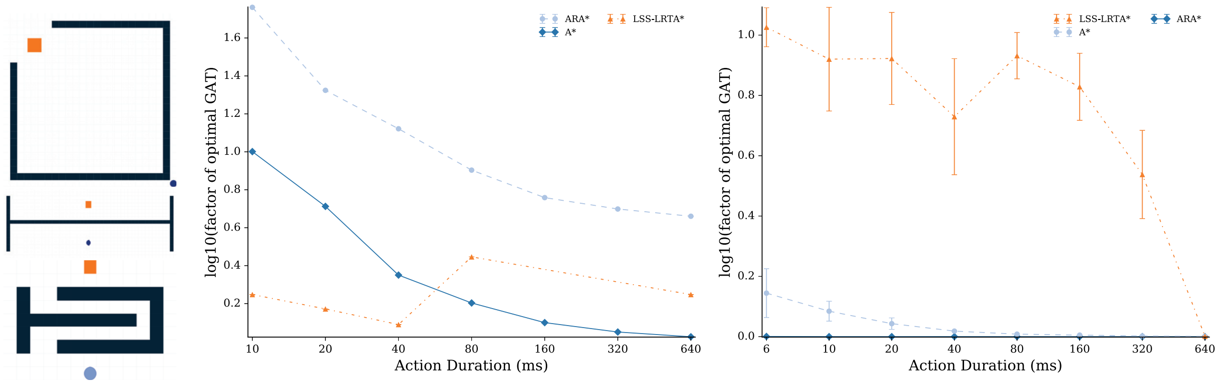

Figure 1: Left: Open box, H-box, and tiny slalom instances. The orange box marks the start and the blue circle the goal state. Middle: GAT on a slalom map with a double integrator. Right: GAT for grid pathfinding in the open box map.

//youtu.be/oPTQuvDFVjU.

Our study found that using real-time heuristic search planning algorithms can have significant benefits, even in complicated domains. In almost all instances in each of the domains used, real-time search outperforms anytime search in terms of goal achievement time (see Figure 1 for example).

The initial delay incurred by anytime search outweighs the typically shorter path to the goal in terms of the overall goal achievement time.

The one major exception to this was the open box instance (Figure 1). In this instance, ARA* greatly benefited from planning backwards from the goal because the start state is in a large heuristic minimum. ARA* should perform well on domains in which heuristic values near the goal are more accurate than near the start and planning backwards is advantageous. However, when the backwards planning advantage was removed by creating a symmetric hbox instance (Figure 1), ARA* performed similarly to the real-time algorithms.

One disadvantage of real-time search is that it is susceptible to dead ends. This was especially evident with a double integrator domain in instances that have long open spaces since real-time search will attempt to increase its velocity as much as possible towards the goal but if there are obstacles between the agent and the goal, the agent may not be able to slow down in time causing a collision.

ARA* could not be run on the acrobot because planning backwards from the goal requires calculating predecessor states, which is non trivial.

Finally, our results showed that, as the action durations are increased, the goal achievement times of the algorithms converge as all of the algorithms have a sufficient time to plan and find good solutions.

Discussion

To our knowledge, this is the first time heuristic search has been used for the robotics-oriented double integrator and Acrobot domains, and the first time that real-time heuristic search has been implemented to follow a strict hard real-time bound. The results show that real-time search generally outperforms anytime search in terms of goal achievement time in the domains we tested. Anytime search is currently the leading heuristic search approach taken in advanced robotics work where time is important, so these results suggest that further investigation of the real-time approach is warranted. The main drawback of real-time search illustrated in our results is their incompleteness in the presence of dead ends; developing methodologies to tie lookahead time to safety given a vehicle's dynamics is a promising area for future work.

Acknowledgments

We gratefully acknowledge support from NSF (award 1150068).

References

Barto, A. G.; Bradtke, S. J.; and Singh, S. P. 1995. Learning to act using real-time dynamic programming. *Artificial Intelligence* 72(1):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* 54:123–158.

Koenig, S., and Sun, X. 2009. Comparing real-time and incremental heuristic search for real-time situated agents. *AAMAS* 18(3):313–341.

Likhachev, M.; Gordon, G.; and Thrun, S. 2004. ARA*: Anytime A* with provable bounds on sub-optimality. In *ANIPS 16*.

Likhachev, M., and Ferguson, D. 2009. Planning long dynamically feasible maneuvers for autonomous vehicles. *IJRR* 28(8):933–945.

Likhachev, M. 2016. Personal communication.

Murray, R. M., and Hauser, J. E. 1991. *A case study in approximate linearization: The acrobat example*. Electronics Research Laboratory, College of Engineering, University of California.

Sturtevant, N., and Bulitko, V. 2014. Reaching the goal in real-time heuristic search: Scrubbing behavior is unavoidable. In *SoCS*, 166–174.