Anticipatory On-line Planning

Ethan Burns[1] **J. Benton**[2] **Wheeler Ruml**[1] **Sungwook Yoon**[3] **Minh B. Do**[4]

[1]Dept. of Computer Science, University of New Hampshire, Durham, NH 03824 USA, eaburns at unh.edu, ruml at cs.unh.edu

[2]Dept. of CS and Eng., Arizona State University, Tempe, AZ 85287 USA, j.benton at asu.edu

[3]Embedded Reasoning Area, Palo Alto Research Center, Palo Alto, CA 94304 USA, sungwook.yoon at parc.com

[4]Planning & Scheduling Group, SGT Inc., NASA Ames Research Center, Moffett Field, CA 94035 USA, minh.b.do at nasa.gov

Abstract

We consider the problem of on-line continual planning, in which additional goals may arrive while plans for previous goals are still executing and plan quality depends on how quickly goals are achieved. This is a challenging problem even in domains with deterministic actions. One common and straightforward approach is *reactive* planning, in which plans are synthesized when a new goal arrives. In this paper, we adapt the technique of hindsight optimization from on-line scheduling and probabilistic planning to create an *anticipatory* on-line planning algorithm. Using an estimate of the goal arrival distribution, we sample possible futures and use a deterministic planner to estimate the value of taking possible actions at each time step. Results in two benchmark domains based on unmanned aerial vehicle planning and manufacturing suggest that an anticipatory approach yields a superior planner that is sensitive not only to which action should be executed, but when.

Introduction

Consider the problem faced by a unmanned aerial vehicle (UAV) dispatcher who must plan for a set of UAVs to service a set of observation requests. To service a request, one of the UAVs must fly over a given strip of land with its observation equipment turned on. The dispatcher wants to minimize the time between when a request arrives and when an UAV has completed the flyover. Even when the actions of the UAV, such as flying particular routes or switching on/off observational equipment, can be regarded as deterministic, the stochastic arrival of new requests can make for a challenging on-line planning problem. What should the UAVs do when no requests are pending? To maximize expected utility, it might be worth preemptively moving them near locations where requests are likely to arrive. A similar problem occurs in situations where significant set-up costs are involved. For example, problems like manufacturing often have relatively predictable actions but the question is whether to immediately plan and execute every time a request arrives. It may be worth the cost of delaying production to wait for more orders, so as to reduce the total cost of execution for the entire batch. We call these *on-line continual planning* problems (OCPPs). They incorporate aspects of planning (action selection), scheduling (when to perform the actions), and uncertainty (unknown future goals).

Traditionally, there are two major strategies for handling the on-line arrival of additional goals. The first is a greedy strategy, in which previous plans are held fixed and plans for the new goals are restricted to be consistent with the old ones. This simple approach was taken by ter Mors, Zutt, and Witteveen (2007) in their work on airport taxiways and by Ruml et al. (2011) in their work on planning for modular printers. The second strategy re-plans for all goals from scratch (Nebel and Koehler 1995; Knight et al. 2001; Fox et al. 2006). Both of these strategies wait to plan until goals actually arrive and consider only those goals that have been revealed. These *reactive planning* approaches neglect any available information about possible future goal arrivals and therefore they fail to preemptively arrange for better handling of future goals, even when they are expected.

In this short paper, we consider an alternative approach, *anticipatory on-line planning*, that explicitly exploits information about future goal arrivals. It assumes that the probability distribution over incoming goals is either known or learn-able and employs the technique of *optimization in hindsight*, previously developed for on-line scheduling and recently investigated for planning with stochastic actions (Mercier and van Hentenryck 2007; Yoon et al. 2008; 2010). This technique first samples from the distribution of possible future goal arrivals and then considers which next action optimizes the expected cost when averaged over the sampled futures. By using this anticipatory technique, our planner is able to take future goals into account. Although anticipatory planning involves more computation than reactive planning, we find that the resulting planner finds plans with much higher expected utility when tested in benchmark domains modeled on the two examples above.

On-line Continual Planning

While most research has concentrated on off-line scenarios, planners deployed for real-world applications are frequently run in an on-line setting in which goal arrival, plan synthesis, and plan execution are concurrent and interleaved. Such domains include manufacturing process control, supply chain management, power distribution network configuration, transportation logistics, mobile robotics, and spacecraft control. Despite the importance of these on-line domains, little attention has been paid to the potential drawbacks of using standard off-line (and therefore reactive)

planners in an on-line environment.

In the problems that we consider here, the world is deterministic, however, new goals may arrive at each time step with some known distribution. The agent must decide which action to perform at each step. We distinguish between the *world state*, the configuration of the world, and *state*, the combination of the world state and the current set of goals that must be achieved. We define the on-line continual planning problem (OCPP) as a Markov decision process (MDP) with both world states and goal sets. An OCPP consists of a 5-tuple $P = \langle W, G, A, T, C \rangle$, where W is a set of world states, G is a set of goals and A is a set of actions. A state $s = \langle w, g \rangle \in S$ is a combination of a world state $w \in W$ and a current goal set $g \subseteq G$, therefore, the set of possible states is as $S = W \times 2^G$. The function $T : S \times A \times S \to \mathbb{R}$ is the transition function: $T(\langle w, g \rangle, a, \langle w', g' \rangle)$ gives the probability of transitioning from a world state w with goal set g to a world state w' with goal set g' when performing action a. Because we consider a deterministic world with stochastic goal arrivals, the T function, can be thought of as defining the goal arrival distribution. This differs from traditional MDP models that do not explicitly define goals or their arrival. The function $C : S \times A \to \mathbb{R}$ defines a cost for performing an action in a given state. Reward is represented as negative cost.

Because the notion of a complete plan is less meaningful in an on-line continual context, we consider the objective of minimizing cost over a horizon H starting from the current state s_1. If a sequence of actions $a = \langle a_1, ..., a_H \rangle$ is performed, giving a sequence of states $s = \langle s_1, ..., s_H \rangle$, then we compute the total cost as $C_H(s, a) = \sum_{i=1}^{H} C(s_i, a_i)$. Note that the horizon is defined over action transitions and, as formulated above, the transition includes both action execution and goal arrival.

Anticipatory On-line Planning

Optimization in hindsight was originally developed in the scheduling and networking communities (Chong, Givan, and Chang 2000; Mercier and van Hentenryck 2007; Wu, Chong, and Givan 2002) and has recently been applied to probabilistic planning (Yoon et al. 2008; 2010). Rather than waiting for goals to arrive or assuming a single 'most likely' future, hindsight optimization samples possible futures according to their distribution. In this way, it incorporates probabilistic information. It then solves each sampled future with a fast deterministic planner, achieving scalability.

As an example, consider the UAV dispatching problem. Given that we have an estimate of the distribution over possible future observation request locations, we can sample possible future sequences of requests. We can then evaluate each available action for a UAV by imagining the average cost of continuing after that action to complete the anticipated, sampled requests. In this way, optimization in hindsight allows us to intelligently anticipate future goal arrivals.

Following Chong, Givan, and Chang (2000), we can define the value of being in a state s_1 as the minimum expected plan cost that extends from s_1. That is, the minimum over all possible future action sequences of the total cost over all expected future states:

$$V_H^*(s_1) = \min_{a_1,...,a_H} \underset{s_2,...,s_H}{E} \left[\sum_{i=1}^{H} C(s_i, a_i) \right]$$

Recall that states are composed of both world states and goal sets. Since we only consider deterministic world state transitions, the expectation over future states $s_2, ..., s_H$ is actually over the distribution of possible goal arrivals. So, given the goal arrival distribution, we would like to find the action sequence $a_1, ..., a_H$ that minimizes the sum of the costs over the expected goal arrivals. To compute V^* exactly, we can perform an expectation minimization search. Unfortunately, this requires us to compute the expectation for each of exponentially many plans in the horizon H. This can be very costly.

Following the principle of optimization in hindsight, we may instead approximate the value function by exchanging expectation and minimization, so that we are taking the expected value of minimum-cost plans instead of the minimum over expected-cost plans. We then have:

$$\hat{V}_H(s_1) = \underset{s_2,...,s_H}{E} \left[\min_{a_1,...,a_H} \sum_{i=1}^{H} C(s_i, a_i) \right]$$

This approximation of $V_H^*(s)$ uses fixed future goals for each minimization, making it much more manageable. For each possible future in the expectation, the problem becomes the deterministic planning problem of finding an action sequence that minimizes cost given a fixed set of future goals, i.e., cost-based deterministic planning. We define a Q-value to be the potential cost of taking an action a_1 in the state s_1:

$$Q(s_1, a_1) = C(s_1, a_1) + \underset{s2,...,s_H}{E} \left[\min_{a_2,...,a_H} \sum_{i=2}^{H} C(s_i, a_i) \right]$$

From this we estimate the best action choice in s_1 as $\min_a Q(s_1, a)$. Using this technique, we are said to be performing optimization with the benefit of 'hindsight' knowledge about how future uncertainty will be resolved.

Figure 1 summarizes our anticipatory planning algorithm. At each time step, the algorithm is used to find the next action to execute for the current state s. First, we generate a set of $Width$ samples, using the $draw\_future()$ function, where each sample is a sequence of future goal arrivals. This function samples a goal from the known distribution for each time step from 1 to H (lines 1–2). Because world state transitions are deterministic, our algorithm samples over the distribution of possible goal arrivals occurring in the upcoming time steps, rather than over possible action outcomes

ANTICIPATORY-PLANNING$(s = \langle w, g \rangle, G, H, Width)$
1. for i from 1 to $Width$ do
2. $\quad f_i \leftarrow draw\_future(G, H)$
3. foreach action a applicable in world state w
4. $\quad s' \leftarrow \langle a(w), g \rangle$
5. $\quad m \leftarrow (\sum_{i=1}^{Width} solve(s', f_i))/\, Width$
6. $\quad Q(s, a) \leftarrow C(s, a) + m$
7. Return $argmin_a\, Q(s, a)$

Figure 1: The anticipatory planning algorithm.

as is done in hindsight optimization for probabilistic planning. Next, for each action, we advance deterministically to world state resulting from action a in w, which we notate as $a(w)$. This gives us a new state $s' = \langle a(w), g \rangle$ that contains the new world state $a(w)$ and the goal set g from the current state (line 4). Then, each possible future f_i is grouped with the state s', generating a deterministic planning problem. Solving this problem provides an optimal solution to the future f_i. The mean optimal solution cost across the set of samples (line 5) along with the cost of the action $C(s, a)$ is used as the Q-value for each action a in state s (line 6). Finally, we return the action with the minimum Q-value (line 7).

Empirical Evaluation

We compared this on-line anticipatory algorithm to a reactive planner on our two example domains: UAV dispatching and production planning. Both the anticipatory and reactive planners used an A* based search, however our approach can easily be adapted to use any off-line, cost-optimal, deterministic planner. We also implemented a greedy planner that always took the action that minimized the sum of the next action's cost and a heuristic estimate. This domain-dependent heuristic was the same as that used in the deterministic search.

Our simulator calls the planner at each time step to determine the next action to perform. After each action is performed, a new goal may arrive according to the known distribution for the given domain. Each instance used in the simulation is characterized by the future goal arrival sequence. We generated these sequences out to 80 time steps. Before simulation, we used an oracle planner to find the optimal plan for each instance. Next, each planner was run in the simulation for a fixed number of simulated time steps equal to 1.25 times the length of the optimal plan.

Unmanned Aerial Vehicle Domain: The UAV domain involves planning the actions of a set of UAVs on a grid with eight-way connectivity in order to minimize the time required to service observation requests. Each observation request is specified by a start and end location. To service a request, a UAV must move to the start location for the request, switch on its instrumentation, traverse to the end location and then shut off its instrumentation. In our experiments, we use a uniform distribution for request begin and end locations. Switching on observation instruments and horizontal and vertical moves of the UAV had a cost of 1 and diagonal moves cost $\sqrt{2}$. An additional penalty of 1 was charged at each time step for each outstanding request that was not being serviced by any UAV, and a penalty of 0.5 was changed for each request that was in the process of being serviced. These penalties promote prompt service of requests. Finally, for each completed request, a reward equal to the length of the request plus the length of the length of the diagonal of the grid was given. This ensures that it is actually worth servicing a request. These rewards were presented as negative costs during the action of switching off observation instrumentation (i.e., when the goal was achieved).

Figure 2 shows the results of the experiment with a set of 100 instances of the UAV domain on a 7×7 grid for 1, 2 and 3 UAVs with observation request arrival probabilities of both 0.02 and 0.04 per time step. The mean optimal solution cost over this set of instances was -22.2 with a standard deviation of 14.8. Each box in the figure surrounds the middle half of the costs, the horizontal line represents the median value, the 'whiskers' extend to the extremes. The gray bars indicate 95% confidence intervals for the mean. The y axis represents the reward achieved on each instance, normalized between the reward obtained by the greedy planner (0) and the optimal oracle planner (1). Any datum lying below the 0 line represents an instance where the planner performed worse than the greedy planner.

From this figure, we can see that the boxes for the reactive planner appear as flat lines at $y = 0$ indicating that the reactive planner often found solutions that were equivalent in reward to the plans found by the greedy solver. While some runs of the anticipatory planner were below the zero line (for any configuration this occurred no more than 14 times and as few as 1 time out of 100 runs), well over 3/4 of the distribution of normalized reward values are significantly better than the greedy solver and the reactive planner in this domain. The solutions found by the anticipatory planner were as close as 52% of the way to optimal on the median and a maximum of 99.999% of the way to optimal. We also see that increasing the sample horizon for the anticipatory planner tended to lead to better solutions. This effect was more pronounced for the more difficult instances with arrival probability 0.04 and multiple UAVs.

In the most difficult setting (3 UAVs, h=8, and prob=0.04), the hindsight planner took a median of 10 seconds to solve each instance, and 0.1 seconds to plan each action. In the easiest setting (1 UAV, h=4, and prob=0.02) it took a median of 0.2 seconds and 0.002 seconds per action. The reactive planner took almost no time to plan each action.

Because we defined an OCPP as a form of MDP, it is natural to ask how an MDP solver would perform on this type of problem. We hypothesized that an MDP solver would not fare well because the state space is so large in these problems. For example, the state space for a 3×3 UAV problem that is restricted to have no more than three outstanding requests with a possible fourth request that is currently being serviced has close to 70 million states. To verify our hypothesis, we implemented a planner based on labeled real-time dynamic programming (Bonet and Geffner 2003) and ran it on a set of 100 3×3 UAV instances. When given approximately two orders of magnitude more time than the reactive and anticipatory planners required on each instance, the LRTDP based planner found plans that were nearly equivalent to those found by the reactive and greedy planners and was outperformed by the anticipatory planner. Because the state space grows very quickly as the grid size increases, this approach will perform much worse on 7×7 grids like the ones used in Figure 2. We omit further results with this approach.

Manufacturing Domain: In contrast to the UAV domain, where preparatory actions are worthwhile, the manufactur-

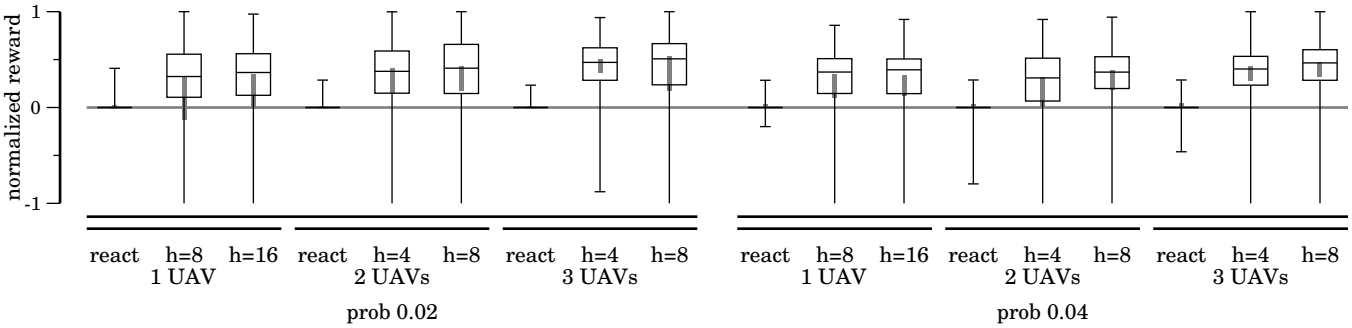

Figure 2: 7×7 UAV instances with request arrival probabilities of 0.02 and 0.04 and 1, 2 and 3 UAVs. The anticipatory planner used 32 samples and horizons of 8 (h=8) and 16 (h=16) with 1 UAV and, 4 (h=4) and 8 with 2 and 3 UAVs.

ing domain models problems where delaying action can lead to lower cost. The goal of this domain is to plan actions for a machine that makes widgets and occasionally requires repair. At each time step, a widget request may arrive that can be satisfied by producing a widget set using the machine. Additionally, any of the parts of the machine may acquire damage. While the machine will still produce widgets regardless of the amount of damage, operating the machine becomes. The amount of damage on a part of the machine is represented by an integer value and the parts of the machine may be fixed by bringing the machine off-line and repairing it entirely.

The available actions transition between the three states of the machine: idle, active, and down for repair. When the machine is in the active state it may produce one set of widgets at each time step. When the machine is down for repair, the repair action may be applied in order to fix all of the damaged parts at once. The cost of each action (producing a widget, transitioning states of the machine and repairing the machine) is 1, there is an additional cost of 1 for each outstanding widget request, encouraging prompt production. Finally, every action is penalized for every part of the machine that has acquired damaged as follows: if the part has acquired n units of damage then the penalty is 1 if $n = 1$ and $\frac{3n}{4}$ otherwise. This cost function reflects the fact that requiring more service on the same part of the machine may be less costly than the initial service requirement. This demonstrates that anticipatory planning can easily handle a complex cost function with goal utility dependencies (Do et al. 2007), where the utility of repairing the machine depends on the service requirement goals for the damaged parts.

Because the repair action fixes all of the damaged parts of the machine at once, the setup cost for bringing the machine down for repair may outweigh the benefits of repairing the machine unless it has acquired a sufficient amount of damage. A successful planner should recognize that it can wait for more parts to break before paying the cost of bringing the machine down for maintenance.

In the manufacturing domain, we used a set of 100 instances and varied the order arrival and damage probabilities between 0.2 and 0.4. The mean optimal solution cost for this set was 254.3 with a standard deviation of 148.9. All planning times in this domain were nearly instant. Figure 3 shows the normalized reward for the reactive planner and the anticipatory planner. We also generated results with a

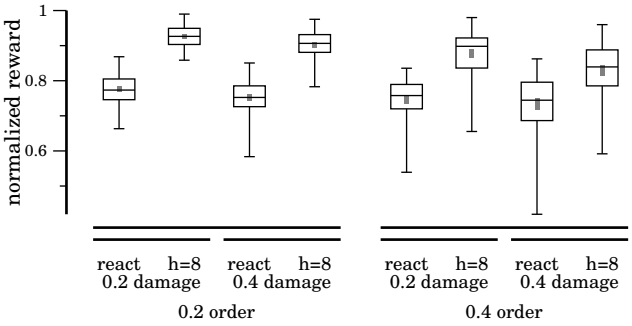

Figure 3: Manufacturing domain using machine with of two parts and order and damage probabilities of 0.2 and 0.4. The anticipatory planner used 32 samples and a horizon of 8.

sample horizon of 16, however, they showed little improvement over the horizon of 8 in this domain. Unlike the UAV domain, the reactive planner gave plans that were closer to optimal than the greedy solver. The anticipatory planner, again, gave a significant improvement over the reactive planner. We can see that on problems with low goal arrival rates, the worst anticipatory planner result was better than the great majority of the the reactive planner results.

Conclusion and Future Work

Recent results have exhibited the power of hindsight optimization (Yoon et al. 2010; 2008) and UCT (Kocsis and Szepesvari 2006; Keller and Eyerich 2011) in planning with probabilistic actions. This paper has formulated the problem of on-line continual planning and shown that Monte Carlo roll-out-based methods can extend easily to this setting, even with complex cost functions. There is much room for future work. It would be interesting to adaptively determine how much time to allocate to the deterministic planner for its search over sampled future goal arrivals. We would also like to investigate the possibility of handling parallel actions.

Acknowledgements

We would like to thank Allen Hubbe, Rao Kambhampati, NSF (grants IIS-0812141 and IIS-0905672), DARPA (grant HR0011-09-1-0021), ONR (grants N00014-09-1-0017 and N00014-07-1-1049), and ARL (contract W911NF-11-C-0037).

References

Bonet, B., and Geffner, H. 2003. Labeled RTDP: Improving the convergence of real-time dynamic programming. In *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS-03)*.

Chong, E.; Givan, R.; and Chang, H. 2000. A framework for simulation-based network control via hindsight optimization. In *IEEE Conference on Decision and Control*.

Do, M. B.; Benton, J.; van den Briel, M.; and Kambhampati, S. 2007. Planning with goal utility dependencies. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*.

Fox, M.; Gerevini, A.; Long, D.; and Serina, I. 2006. Plan stability: Replanning versus plan repair. In Long, D.; Smith, S. F.; Borrajo, D.; and McCluskey, L., eds., *ICAPS*, 212–221. AAAI.

Keller, T., and Eyerich, P. 2011. Prost. In *International Probabilistic Planning Competition 2011 (IPPC 2011)*.

Knight, R.; Rabideau, G.; Chien, S.; Engelhardt, B.; and Sherwood, R. 2001. Casper: Space exploration through continuous planning. *IEEE Intelligent Systems* 16(5):70–75.

Kocsis, and Szepesvari. 2006. Bandit based monte-carlo planning. In *European Conference on Machine Learning (ECML-06)*.

Mercier, L., and van Hentenryck, P. 2007. Performance analysis of online anticipatory algorithms for large multistage stochastic programs. In *International Joint Conference on Artificial Intelligence*.

Nebel, B., and Koehler, J. 1995. Plan reuse versus plan generation: A theoretical and empircal analysis. *Artificial Intelligence* 76:427–454.

Ruml, W.; Do, M. B.; Zhou, R.; and Fromherz, M. P. 2011. On-line planning and scheduling: An application to controlling modular printers. *Journal of Artificial Intelligence Research (JAIR)* 40:415–468.

ter Mors, A. W.; Zutt, J.; and Witteveen, C. 2007. Context-aware logistic routing and scheduling. In *Proceedings of the International Conference on Automated Planning and Scheduling*.

Wu, G.; Chong, E.; and Givan, R. 2002. Burst-level congestion control using hindsight optimization. *IEEE Transactions on Automatic Control*.

Yoon, S.; Fern, A.; Givan, R.; and Kambhampati, S. 2008. Probabilistic planning via determinization in hindsight. In *Proceedings of Conference on Artificial Intelligence (AAAI)*.

Yoon, S.; Ruml, W.; Benton, J.; and Do, M. B. 2010. Improving determinization in hindsight for on-line probabilistic planning. In *Proceedings of the Tenth International Conference on Automated Planning and Scheduling (ICAPS-10)*.