Buying Time for Resource-Bounded Planning

Ella M. Atkins **Edmund H. Durfee** **Kang G. Shin**

University of Michigan
143 ATL, 1101 Beal Ave.
Ann Arbor, MI 48109-2110
{marbles, durfee, kgshin}@umich.edu

ABSTRACT

To react appropriately in hard real-time environments, a system must either employ fast dynamic planning or recall a pre-built reactive response. For complex problems, real-time, accurate dynamic planning may be impossible, and the complete set of possible reactions may be too large to build and store in advance. In this paper, we propose combining offline and online planning such that a set of offline plans are built and stored to maximize the amount of time available for any subsequent online planning. We explore this concept in CIRCA, the Cooperative Intelligent Real-time Control Architecture, which has concentrated on guaranteeing failure avoidance via a combination of planning, scheduling, and real-time plan execution. We are in the process of implementing a plan cache which will increase CIRCA's overall likelihood of success in time-constrained situations, and we illustrate potential gains from these modifications using a simple example from the fully-automated aircraft domain.

Introduction

Achieving safe, fully-automated control of a dynamic system requires fast, accurate responses to maintain safety while also driving the system toward its objectives. Two basic approaches to this problem include using prebuilt reactive plans and employing online planning. Ideally, a comprehensive set of plans could be built and scheduled offline, allowing the system to precompute and guarantee its response time to any runtime situation. However, a prebuilt set of reactive plans may not provide appropriate responses to all possible situations, particularly for complex problems in which domain knowledge may be either incomplete or imprecise. Conversely, online planning may be used to search for appropriate reactions to situations as they arise, but online deliberation must be bounded such that it terminates before the available resource limits are exceeded.

We are concerned with the implications of requiring hard real-time response guarantees, particularly with respect to limiting deliberation time. Many approaches have been used to bound planning time, including anytime (Dean et al., 1993), design-to-time (Garvey and Lesser 1993), and abstraction planning (Boutilier and Dearden 1994) algorithms. For example, an anytime planner will build its solution iteratively such that it may provide the best solution it has computed when interrupted. Generally, the quality of an anytime planning result increases as available deliberation time increases, and some minimum deliberation time (dependent on the problem domain) is required before the anytime planner can generate a result that is better than a simple random guess. Similar trends for deliberation time vs. solution quality are present for the design-to-time and abstraction approaches. Thus, regardless of which algorithm is used, it is advantageous to maximize the amount of deliberation time available to the system, and crucial to allow at least the minimum amount of time required to build a minimally acceptable plan.

In this paper we discuss a hybrid system that combines offline and online planning, and examine how offline reasoning can maximize the amount of time available for any necessary online planning. We first discuss our approach, in which all time bounds are computed from hard real-time reaction deadlines that must be met in order to avoid any "transition to failure" (such as crashing the automated system). During offline deliberations, these time bounds are computed and used to specify a minimal set of plans which must be built offline to *guarantee* avoidance of the most probable failure situations that may be reached. Next, we discuss the ongoing implementation of these algorithms in the Cooperative Intelligent Real-time Control Architecture (CIRCA) (Musliner, Durfee, and Shin 1995), a system which combines methods from both the real-time and planning fields to allow real-time response guarantees in time-critical situations. We illustrate how "buying time" for planning may improve system performance in the context of fully-automated aircraft flight control, then discuss the benefits and possible limitations of the hybrid architecture, including challenges we are currently facing in our CIRCA implementation.

Approach: Require Online Planning only when Time is Available

We propose an architecture which combines offline and online planning, using offline deliberations to develop reactions to the most time-critical situations and online planning to react when the set of existing plans is insufficient. In this section, we first describe how a planner might compute available deliberation time from any world state using the notion of "failure avoidance", and present our definition of "planned-for" vs. "unplanned-for" states. Next, we describe how we combine offline and online planning to allow large state spaces while also enabling fast reaction times when necessary.

For this work, we assume a planner places primary importance on avoiding all possible catastrophic failures and gives secondary consideration to goal achievement, since a system will no longer function in any capacity if it "fails". To operate in this manner, the planner must be capable of identifying states that represent failure, and further, the planner must be capable of computing the amount of time the system will remain "safe" in each state it encounters, where we define a state as "safe" while it cannot transition to failure. Many failure transitions cannot occur until some minimal amount of time passes after a state is first reached. For example, with an aircraft in flight, this time may correspond to the minimum delay between first reaching state "collision-course traffic appears on aircraft radar" and failure state "aircraft crashes".[1] To achieve guaranteed real-time response in such a system, this minimum delay must correspond with the maximum deliberation time available before the system must select and complete its failure avoidance reaction.

The final product of a planning system, the plan, contains reactions to handle only a subset of all possible world states, since it is infeasible to assume the existence of a universal plan set in complex problem domains (Ginsberg 1989). We consider the set of states for which appropriate reactions have been prepared the "planned-for" set, and any other states "unplanned-for". As discussed in (Atkins, Durfee, and Shin 1997), a planner can use its model of the world to detect transitions to "unplanned-for" states, a particularly crucial endeavor when such states may lead to system failure.

Our focus in this paper is to illustrate how careful offline reasoning can augment the quality of online, time-constrained responses whenever a system departs from the set of "planned-for" states. Before the system "starts" (i.e., acts in its environment), offline planning is used to develop the knowledge (or plans) necessary to make online planning and plan retrieval more efficient. In our work, we allow offline planning to be dependent on whatever initial state and goals are present for the current problem instantiation, so we cannot simply dismiss such operations as part of permanent knowledge base development. For example, consider a fully-automated commercial airliner preparing for a flight. A knowledge base would be huge if it contained all plans to *safely* fly from all airports to all others. If, instead, it used the current departure point (initial state) and destination (goal state), offline planning will allow the system to build and store only plans associated with the upcoming flight, not all possible flights across the world. Such a plan could have the appropriate detail required for quick response to a variety of situations, based on features such as local geography and airport traffic patterns.

Consider the situation in which a certain plan, developed offline, begins execution. So long as the states actually reached have been explicitly handled (or planned-for), no online plan modifications or additions are required. However, consider the case where the environment deviates from this set of "planned-for" states, illustrated by the temporal transitions (*tt*s) leading out of "Planned-for States 1" in Figure 1. We assume such an anomalous (unplanned-for) state will be quickly detected, then the system must identify the proper system response. Our criterion for selecting system response is based on one quantity: the minimum delay before the system may transition to failure, traversing one of the "*ttf*" links in Figure 1. We define a "fast" temporal transition to failure (*ttf*) as a transition which occurs so quickly that none of the available online planning techniques may find adequate responses. For such situations, it is important to have pre-planned reactions, so part of the "knowledge" developed during offline planning must include reactive plans that can be retrieved to handle such unplanned-for states. Otherwise, the system adopts a meta-level design-to-time approach in which different planning algorithms are invoked based on *ttf* delay. For example, as illustrated in Figure 1, unplanned-for states are handled via classification as "fast", in which case pre-planned reactions exist, "slow", in which case a moderate-speed planning algorithm such as case-based reasoning may be selected, or "very slow" (including non-existent), in which case a full state-based planner may be employed. Of course, each planning algorithm must be guaranteed to either terminate or be interruptible before the available deliberation time expires.

[1] The notion of "minimum delay" before a temporal transition to failure (*ttf*) is discussed further in (Musliner, Durfee, and Shin 1995).

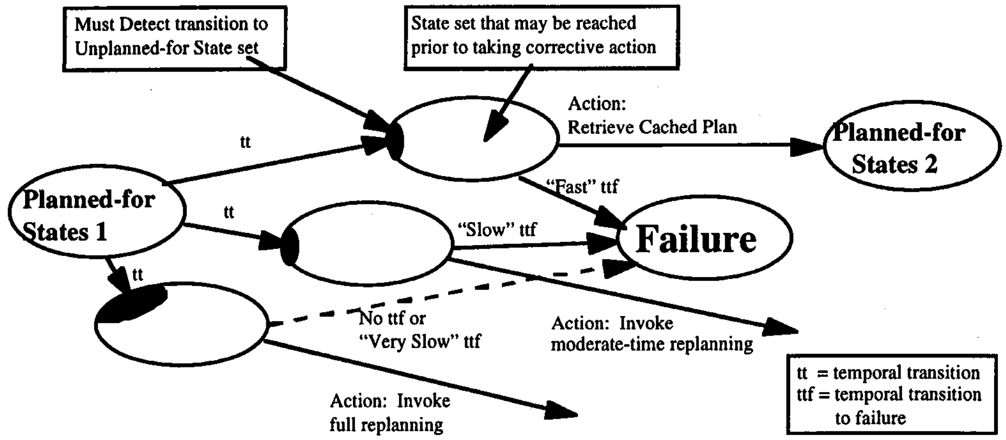

Figure 1. "Plan-space" Transitions based on Time to Failure.

Regardless of the specific planning algorithms employed, the key aspect of our approach lies in the offline computation of problem-specific knowledge that may be used during online deliberation. As discussed above, pre-planned reactions to preempt "fast" *ttfs* must be computed offline. Additionally, depending on the planning algorithms used, offline planning may facilitate other planning operations, such as the building a high-level case library, or reorganizing domain knowledge for more efficient access depending on the goals to be achieved. We explore a specific instantiation of our approach below.

Implementation in CIRCA

The Cooperative Intelligent Real-time Control Architecture (CIRCA) is designed to provide guarantees about system performance with limited resources. Based on a user-specified domain knowledge base, CIRCA's main goal is to build a set of plans that keep the system "safe" while working to achieve its goals if possible. Figure 2 provides a high-level view of the CIRCA architecture we are in the process of completing.[2] The planner searches based on the specified initial state (s) and state transitions to develop a plan of action that will both avoid failure and attempt to achieve the desired goals. This plan, composed of a set of test-action pairs (TAPs), is scheduled such that all time-critical actions will be guaranteed to execute before their deadlines, which correspond to the minimum time before a transition to failure can occur from any state expanded during that planning cycle. Scheduled plans are stored in the plan cache, then downloaded to the real-time plan executor when they are needed.

We are in the process of implementing the plan cache and working out the details of making tradeoffs between cache size and online planning required. Our implementation has begun with the following simple procedure for constructing the initial set of cache plans. First, for each subgoal to be achieved, CIRCA builds a "nominal" plan (depicted by "Planned-for States 1" in Figure 1) offline which will be executed first. This plan will be sufficient if all goes normally, i.e., only the highest-probability "expected" events occur. These nominal plans also contain tests to detect when "unplanned-for" events occur, as described in (Atkins, Durfee, and Shin 1997). For the set of unplanned-for states that may quickly transition to failure (those with an outgoing "fast *ttf*" in Figure 1) a contingency plan is required. CIRCA builds a set of these quick-reaction plans offline, and minimizes the size of each contingency plan by including only failure avoidance as a goal. When the offline planning is complete, CIRCA begins executing the first nominal plan, and if any unplanned-for state is reached, CIRCA retrieves a cached contingency plan if one is available.[3] If no appropriate cached plan exists, CIRCA must invoke online planning with a time bound corresponding to the speed of the "slow *ttf*" or "very slow *ttf*" (illustrated in Figure 1) to build an appropriate reaction.

[2] Previous versions of CIRCA, (Musliner, Durfee, and Shin 1995) and (Atkins, Durfee, and Shin 1997), built and scheduled one plan at a time, then began executing that plan before constructing any other plans. The architecture we discuss here augments CIRCA with the incorporation of a "plan cache" and "resource bounds" on planning processes.

[3] As discussed above, contingency plan reactions are designed to prolong the transition to failure, not necessarily achieve the system goals. To subsequently achieve the goals, the planner may then be invoked, with a longer deliberation time bound than if no contingency plan existed, to build actions that will lead the system back to the goal path.

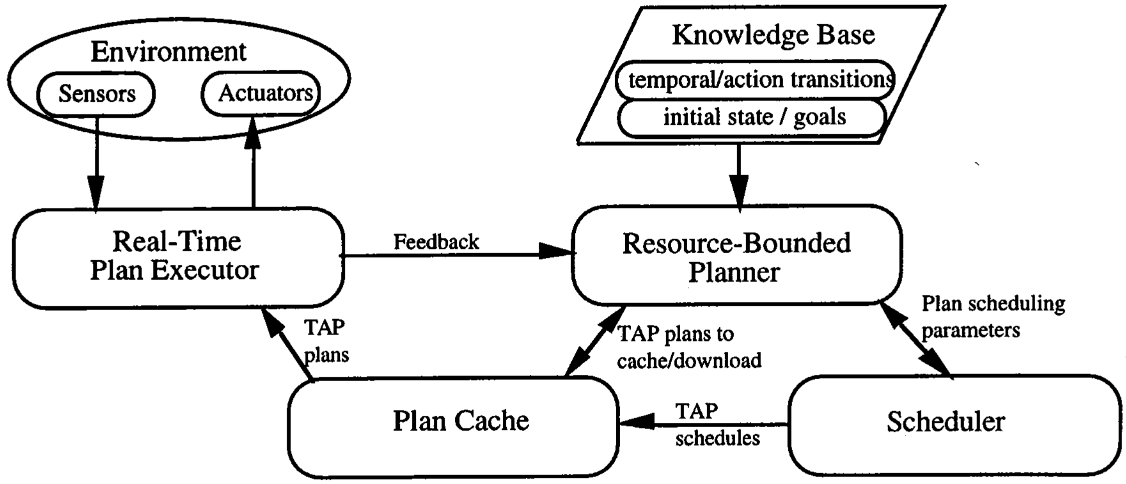

Figure 2. CIRCA Architecture.

Planning and Aircraft Flight

We have begun testing CIRCA's capabilities to build and execute plans that achieve safe, fully-automated aircraft flight, a domain in which real-time demands are crucial because there is no absolutely "safe" state while the plane is airborne. To illustrate how our additions to CIRCA will "buy time" for planning, consider a situation in which the aircraft is on final approach to a normal landing, but senses inoperative landing gear. Because this is an improbable thus "unplanned-for" state (Atkins, Durfee, and Shin 1997), the system leaves the nominal state set for which the executing CIRCA plan will act in guaranteed real-time. Also, since the system fails (crashes) if the aircraft continues its normal descent to landing without gear, there is a "fast" TTF present. Using the proposed system, CIRCA will detect this state and retrieve a cached contingency plan which, in this case, reacts with a "go-around" plan, during which the aircraft climbs and circles around the airport pattern before attempting to complete its landing. The "go-around" effectively avoids the impending crash, but does not achieve the "goal" of safely landing on the runway. Note that without the "go-around" contingency plan, CIRCA would have required a complete online planning cycle with time bound set to the time remaining before the aircraft attempted to touch down on the runway. If the aircraft is close to the runway when gear failure occurs, reacting exclusively by replanning would not be able to offer an alternative action before the crash. With the cached go-around plan, the plane will remain airborne much longer, effectively "buying time" for the planner to select appropriate reactions to the indicated gear problem (e.g., cycling the gear controls, extending gear by backup means, or dumping fuel and preparing for a gear-up landing).

Conclusions

We have outlined a procedure in which offline planning is used to precompile reactions (or more generally, knowledge) to "buy time" for online planning whenever it is required. Reaction time bounds are computed using the notion of preempting transitions to failure from one or more "planned-for" states. These bounds are used offline to build plans for the most time-critical situations and online to control execution time for any required deliberation. The use of these concepts is considered in the context of CIRCA and related to an example in fully-automated aircraft flight.

There are many benefits to the combined use of offline and online deliberation. Offline planning allows a system to build reactions that will provide definite guarantees for avoiding failure in many situations, a claim which is difficult to make when more complex, "unscheduled" planning processes must become involved online. Conversely, online deliberation allows a planner to reason about the full set of "unexpected" situations, an unrealistic expectation of exclusively reactive systems. By combining the two, our system requires online deliberation only when failure is not temporally imminent, which will increase the odds that deliberation may proceed until an acceptable plan is developed.

Our approach contains certain drawbacks, such as the overhead associated with computing each planning time bound and deciding which "planning reaction" is appropriate, for example, retrieving a plan vs. beginning online deliberation in CIRCA. We believe our procedure to build time-critical reactions offline will improve the average-case system performance. However, the worst-case behavior (e.g., large diverse state-space possible and most states lead to failure quickly) may not be better than with either exclusively offline or online planning, and, in fact, may potentially be worse due to the overhead associated with selecting the appropriate "planning reaction".

We subscribe to the philosophy that planning time bounds should be made as large as possible. For this reason, we do not limit deliberation time based on requirements for goal achievement, but rather only on the avoidance of catastrophic failure, a "goal" that absolutely must be achieved if the system is to survive. This strategy is advantageous because nearly all planning systems produce better quality solutions as the available deliberation time increases. However, in situations where a goal becomes unachievable as time passes, a system adopting our strategy will avoid catastrophic failure but may never reach this goal. Conversely, if the system had limited deliberation time based on goal reachability in addition to failure avoidance, the planner may have been able to provide an acceptable (if approximate) solution. We hope to study this tradeoff in more detail, particularly as new algorithms to assess plan quality vs. deliberation time become available.

Acknowledgments

This research was supported under NSF Grant IRI-9209031.

References

Atkins, E. M.; Durfee, E. H.; and Shin, K. G. Detecting and Reacting to Unplanned-for States, to appear in *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, July 1997.

Boutilier, C. and Dearden, R., "Using Abstractions for Decision-Theoretic Planning with Time Constraints," *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pp. 1016-1022, 1994.

Dean, T.; Kaelbling, L. P.; Kirman, J.; and Nicholson, A. Planning with Deadlines in Stochastic Domains. *Proceedings of AAAI*, pp. 574-579, July 1993.

Garvey, A. J., and Lesser, V. R., "Design-to-time real-time scheduling," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 23 no. 6, pp. 1491-1502, 1993.

Ginsberg, M. L. Universal Planning: An (Almost) Universally Bad Idea. *AI Magazine*, vol. 10, no. 4, 1989.

Musliner, D. J.; Durfee, E. H.; and Shin, K. G. World Modeling for the Dynamic Construction of Real-Time Control Plans. *Artificial Intelligence*, 74:83-127, 1995.