Safe Learning in Mission-Critical Domains: Time is of the Essence

(Extended Abstract)

David J. Musliner
Automated Reasoning Group
Honeywell Laboratories
3660 Technology Drive
Minneapolis, MN 55418
musliner@htc.honeywell.com

Introduction

The central claim this paper makes is that if we are concerned about "safe learning," we cannot ignore issues of timeliness in both the learned behaviors and in the learning process itself. Most existing research ignores both.

The very term "safe learning" implies that there is a way to be unsafe, to do something wrong. And if we are willing to actually spend effort to ensure we'll stay safe, then unsafe is probably pretty bad. Hence "safe learning" seems aimed towards the problems of developing autonomous control systems for mission-critical domains, where failure to accomplish or maintain mission goals may result in catastrophic, unacceptable forms of failure (e.g., loss of life, large costs). Examples of mission-critical domains include control of autonomous vehicles (e.g. UAVs), semi-autonomous vehicles (e.g., commercial aircraft), and industrial plants (e.g., oil refineries, power plants). Control systems for these types of applications are typically subject to rigorous testing and certification regimes to ensure predictable, correct, and timely behavior. As Stankovic (1988) notes, "In real-time computing the correctness of the system depends not only on the logical result of the computation but also on the time at which the results are produced." Autonomous planning and control architectures that ignore the issue of response timeliness cannot be applied in mission-critical applications.

Furthermore, if a mission-critical domain uses an adaptive or learning control system, then the adaptation process itself may be subject to mission-critical timing and correctness requirements. We believe there are essentially three different types of adaptation (which may all co-exist in a particular domain): non-critical adaptation, postponable adaptation, and real-time adaptation. We are exploring some of these adaptation forms in the Cooperative Intelligent Real-Time Control Architecture (CIRCA) architecture (Musliner, Durfee, & Shin 1993; 1995). Before describing these forms in more detail, we provide a brief review of CIRCA concepts to set the stage.

CIRCA Summary

As illustrated in Figure 1, CIRCA is an architecture for real-time control that combines three functional modules operating in parallel. The Real-Time Subsystem (RTS) reactively executes predictable real-time control plans that sense the state of the world and respond with safety-preserving and goal-achieving actions. The Controller Synthesis Module (CSM) dynamically constructs the reactive control plans to be executed by the RTS. The Adaptive Mission Planner (AMP) is responsible for dividing the overall mission's state-space of possible worlds into smaller, intersecting "regions of competency" (see Figure 2), each of which can be covered by a single automatically-synthesized reactive control plan. The AMP tasks the CSM to create these new control plans both in advance of mission start and on the fly, as conditions change. This online controller synthesis (or planning) provides self-directed adaptation. CIRCA does not yet incorporate other forms of learning such as parameter tuning or learning new models of unexpected world dynamics.

Three Forms of Adaptation

We distinguish three types of potential adaptation in mission-critical application domains:

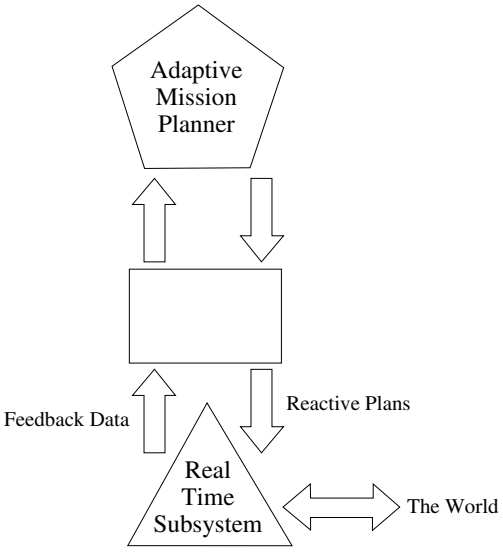

Figure 1: The CIRCA architecture combines intelligent planning and adaptation with real-time performance guarantees.

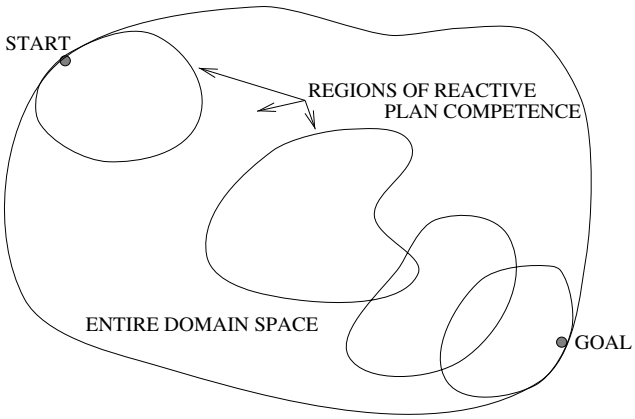

Figure 2: Conceptual view of multiple reaction plans.

Non-Critical Adaptation (Optimization) — Non-Critical Adaptations to the control system (plan) are able to improve performance, reduce costs, or otherwise optimize system behavior, but do not affect the fundamental correctness and timeliness of the mission-critical aspects of system behavior. Since these adaptations do not affect safety-critical aspects, the adaptation process itself is not required to meet timing constraints. In a CIRCA domain, we might have a learning system that tunes the estimates of environmental timing characteristics, but does not change the worst-case timing bounds provided to CIRCA. Better tuning of the timing expectations could allow the system to improve its planning for average-case situations, leading to an overall improvement in goal achievement for nominal executions. However, the worst-case time bounds would not be altered by learning, and hence the plan's fundamental guaranteed real-time reactions would remain unchanged. Since the performance guarantees are unaffected, the system can make this adaptation at any time.

Postponable Adaptation — Postponable Adaptations are critical to the correct and timely behavior of the system in some region of the domain space, but the control system can avoid entering that region until it has finished preparing for the appropriate adaptations. In this case, the adaptation process itself (learning, planning, or controller synthesis) need not meet hard real-time deadlines, but the resulting adapted control system must still provide hard real-time performance guarantees. CIRCA was explicitly designed to support this type of adaptation by building real-time control plans that are guaranteed safe, controllable, and closed (Musliner, Durfee, & Shin 1995). That is, the reactive controllers CIRCA builds are designed explicitly to keep the world safe in a particular region of the state space *and to keep the world* in *that region*, until a new controller is ready for the next state space region. This isolates the adaptation/planning process from the domain's real-time requirements. Of course, this is only possible in domains that permit some type of "holding pattern," so that the current controller can keep the world safe while waiting for the next controller to be created.

Real-Time Adaptation — Finally, the most difficult "safe learning" situation is when the control sys-

tem must adapt or learn to remain safe, and it must do so within hard real-time constraints. In other words, the domain requires the learning process itself to meet hard real-time deadlines, as well as producing a modified control system that meets hard real-time deadlines and ensures system safety. For example, a UAV may be flying towards its destination when an equipment failure occurs, requiring a new reactive plan in a very short time period. If fuel constraints or other mission restrictions make this requirement non-postponable, then the adaptation must occur by the deadline or the mission may fail. In the CIRCA model, this corresponds to having the CSM create new plans on the fly under deadline constraints. We are currently extending the architecture to handle this type of constraint, using deliberation scheduling concepts to manage the adaptation (planning) process and tailor the complexity of the CSM planning tasks to the expected available time (Goldman, Musliner, & Krebsbach 2001).

We believe that these different types of adaptation in mission-critical domains provide a useful perspective for clarifying what aspects of learning must be safe. In particular, the third type of adaptation makes clear the crucial role of timeliness in the adaptation or learning process itself.

The closest related work we are aware of is Gordon's (2001) discussion of "adaptive, predictable, timely" (APT) agents. However, this work addresses only part of the problem of predictable and timely behavior. In particular, the plans that Gordon's APT agents reason about have no temporal semantics, and only the timeliness of the re-verification process (which assures plan safety in strict logical form) is dealt with directly. The timeliness of the overall learning process itself is not addressed, so real-time performance guarantees are not available either for the plans themselves, or the process by which they are adapted. In contrast, CIRCA builds reactive control plans that explicitly account for the timing constraints of the domain, and our recent work is bringing the online controller synthesis (adaptation) process itself under real-time control, to yield more predictable results for the adaptation process.

Acknowledgments

This material is based upon work supported by DARPA/ITO and the Air Force Research Laboratory under Contract No. F30602-00-C-0017. Any opinions, findings and conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA, the U.S. Government, or the Air Force Research Laboratory.

References

Goldman, R. P.; Musliner, D. J.; and Krebsbach, K. D. 2001. Managing online self-adaptation in real-time environments. In *Proc. Second International Workshop on Self Adaptive Software*.

Gordon, D. F. 2001. APT agents: Agents that are adaptive, predictable, and timely. In *Proc. First Goddard Workshop on Formal Approaches to Agent-Based Systems*.

Musliner, D. J.; Durfee, E. H.; and Shin, K. G. 1993. CIRCA: a cooperative intelligent real-time control architecture. *IEEE Trans. Systems, Man, and Cybernetics* 23(6):1561–1574.

Musliner, D. J.; Durfee, E. H.; and Shin, K. G. 1995. World modeling for the dynamic construction of real-time control plans. *Artificial Intelligence* 74(1):83–127.

Stankovic, J. A. 1988. Misconceptions about real-time computing: A serious problem for next-generation systems. *IEEE Computer* 21(10):10–19.