Working Notes of the AAAI Fall Symposium on Negotiation Methods for Autonomous Cooperative Systems
November 2-4, 2001, North Falmouth, Massachusetts

You Sense, I'll Act: Coordinated Preemption in Multi-Agent CIRCA

Kurt D. Krebsbach and David J. Musliner
Automated Reasoning Group
Honeywell Technology Center
3660 Technology Drive
Minneapolis, MN 55418
{krebsbac,musliner}@htc.honeywell.com

Abstract

We are extending the real-time performance guarantees provided by CIRCA into distributed multi-agent systems. In particular, we are developing methods for teams of CIRCA agents to build coordinated plans that include explicit runtime communications to support distributed real-time reactivity to the environment. These teams will then build plans in which different agents use their unique capabilities in a coordinated fashion to guarantee system safety, enabling the application of CIRCA to mission-critical domains that are too hazardous for competing multi-agent approaches.

Introduction

We are extending the existing Cooperative Intelligent Real-Time Control Architecture (CIRCA) for real-time planning and control (Musliner, Durfee, & Shin 1993; 1995) into distributed applications such as the control of multiple unmanned aerial vehicles (UAVs). In such coarse-grain distributed applications, multiple autonomous agents are each controlled by a CIRCA system, cooperating to achieve team goals in mission-critical domains. We are particularly interested in extending the real-time performance guarantees that CIRCA provides for single agents to small teams of coordinating CIRCA agents. This paper discusses the simplest form of coordinated real-time plans of interest: *coordinated preemption*.

Individual CIRCA agents make guarantees of system safety by automatically building reactive control plans that guarantee to *preempt* all forms of system failure. By *preempt*, we mean that an action is planned to disable the preconditions of a potential failure, and that the action is time-constrained to *definitely* occur before the failure could *possibly* occur. Figure 1 provides an example of a preemption in which the agent will take the action of activating electronic countermeasures (ECM) to defeat a missile that is tracking it. If the system has guaranteed to detect a state in which (Missile-tracking T) holds, and perform a responsive action before the missile can hit it, then this action preempts the temporal transition to the failure state. System safety is guaranteed by planning actions that preempt *all* failures (Musliner, Durfee, & Shin 1995).

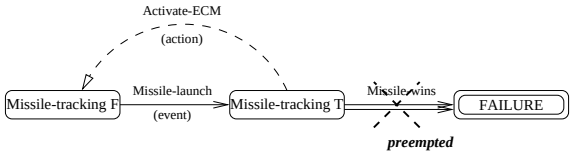

Figure 1: A simple single-agent, single-action preemption example.

By *coordinated preemption*, we mean a set of plans that can be executed by distributed agents to detect, react to, and defeat a threat. For example, suppose one UAV has a sensor that can detect a missile that has been launched at the team, but it has no countermeasures to defeat a missile (perhaps because of weight/power constraints or earlier damage). Furthermore, suppose that another UAV has the appropriate ECM to defeat the threatening missile, but it has no threat-detection sensors. How can the two distributed agents build their plans to accomplish a coordinated preemption: "**You sense** the threat and **I'll act** to defeat it"?

A key observation is that this really devolves into

two separate issues:

Planned communication — The agents must recognize the need to explicitly communicate (both sending and receiving) at a rate fast enough to satisfy the coordinated preemption timing constraint. In our example, the sensing agent must agree not only to detect the missile fast enough, but also to tell the other agent about the threat soon enough. Likewise, the acting agent must focus sufficient attention on "listening" for a message from the sensing agent often enough that it can guarantee to both receive the message and react to defeat the threat, all before the deadline.

Distributed causal links — The distributed agents must be able to represent and reason about changes to their world that are not directly under their control, but which are predictable enough to be relied upon for a preemption guarantee. For example, in our scenario, the sensing agent must rely on the acting agent to take the appropriate action in time to guarantee to detect the missile threat.

Brief Overview of CIRCA

CIRCA uses a suite of planning and scheduling components to reason about high-level problems that require powerful but potentially unbounded AI methods, while a separate real-time subsystem (RTS) reactively executes the automatically-generated plans and enforces guaranteed response times (Musliner, Durfee, & Shin 1993; 1995; Musliner *et al.* 1998). The Adaptive Mission Planner (AMP) and Controller Synthesis Module (CSM) cooperate to build executable reaction plans that will assure system safety and attempt to achieve system goals when executed by the RTS.

CIRCA's planning and execution subsystems operate in parallel. As illustrated in Figure 2, the Controller Synthesis Module reasons about an internal model of the world and dynamically programs the RTS with a planned set of reactions. While the RTS is executing those reactions, ensuring that the system avoids failure, the other system components continue planning to find the next appropriate set of reactions. The derivation of this new set of reactions does not need to meet a hard deadline, because the reactions concurrently executing on the RTS will continue handling all situations, maintaining system safety. When the new

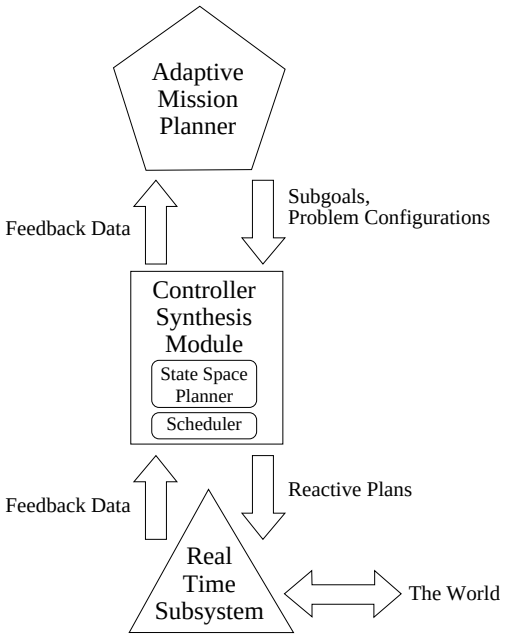

Figure 2: The CIRCA architecture combines intelligent planning and adaptation with real-time performance guarantees.

reaction set has been developed, it can be downloaded to the RTS.

CIRCA's planning system builds reaction plans based on a world model and a set of formally-defined safety conditions that must be satisfied by feasible plans (Goldman *et al.* 1997). To describe a domain to CIRCA, the user inputs a set of transition descriptions that implicitly define the set of reachable states.

The CSM reasons about transitions of four types:

Action transitions represent actions performed by the RTS. These parallel the operators of a conventional planning system. Associated with each action is a worst case execution time (*wcet*): an *upper bound* on the delay ($\Delta(a) \leq T$) before the action completes.

Temporal transitions represent uncontrollable processes, some of which may need to be preempted. Associated with each temporal transition is a *lower bound* on the delay before the temporal transition could possibly occur ($\Delta(\mathrm{tt}) \geq T$). Transition delays with a lower bound of zero are referred to as **Events**, and are handled specially for efficiency reasons.

Reliable temporal transitions represent continuous processes that may need to be employed by the CIRCA agent. For example, a CIRCA agent might

```
#<Single agent TAP>
 Tests: (MISSILE-TRACKING T)
 Acts : ACTIVATE-ECM
```

Figure 3: Single agent Test-Action Pair to activate ECM when a missile is tracking the agent.

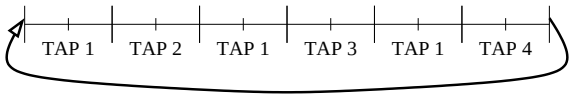

Figure 4: A TAP schedule loop in which TAP 1 must be run more often than the others.

turn on a piece of equipment to initiate the process of warming up that equipment. The action itself will take a relatively short period of time to complete, however, the warming process might complete after a much longer delay. Reliable temporal transitions have both upper and lower bounds on their delays. As we will see, reliable temporals are especially important for modeling multi-agent interactions.

The Controller Synthesis Module builds plans by generating a nondeterministic finite automaton (NFA) from these transition descriptions. The CSM assigns to each reachable state either an action transition or `no-op`. It selects actions to drive the system towards states that satisfy as many goal propositions as possible and to preempt transitions that lead to failure. Action assignments determine the topology of the NFA (and so the set of reachable states): preemption of temporal transitions removes edges and assignment of actions adds them. This ability to build plans that guarantee the correctness and timeliness of safety-preserving reactions makes CIRCA suited to mission-critical applications in hard real-time domains. CIRCA has been applied to real-time planning and control problems in various domains including mobile robotics and simulated autonomous aircraft (Musliner, Durfee, & Shin 1993; 1995; Atkins, Durfee, & Shin 1996).

Coordinated Preemption

Figure 1 illustrates a simple preemption in CIRCA. Figure 3 shows the Test-Action Pair (TAP) automatically generated and scheduled by CIRCA to implement this preemption.

Each TAP has an associated worst-case execution time (*wcet*), which includes the worst-case time to complete the test plus the maximum amount of time to complete the action (if the condition is true). The CIRCA scheduler then uses its world model to derive the maximum allowable response time before a failure could possibly occur. Based on this and the *wcet*, it computes how often the given TAP must be executed to guarantee preempting transition to a failure state.

The CIRCA scheduler then attempts to build a cyclic schedule that runs each TAP at least as frequently as required. It is crucial to preemption that the maximum response time be strictly shorter than the minimum time for one of the undesirable transitions to occur. Figure 4 provides an example cyclic schedule in which TAP 1 must be run more often than the other TAPs. If the scheduler cannot build a satisfactory polling loop, the problem is overconstrained, and the planner must backtrack in its search to compute a feasible plan.

In this paper, we are interested in extending CIRCA to handle preemptive plans that require at least two CIRCA agents to execute. But what does it mean to spread a preemption over two agents? Imagine our original example: "You sense, I'll act". Whereas in single agent CIRCA, both parts would be encapsulated within a single TAP, the test now belongs to one agent, and the action to the other, implying at least one TAP for each agent. But for the two agents to preserve the original semantics of the preemption, they will have to communicate, and that communication will also have occur in a predictable and timely manner.

Thus, a coordinated preemption involves explicit communication between two agents, in this case, one sensing and one acting. To get the CSMs to plan this communication explicitly, the Adaptive Mission Planner (AMP) (Goldman, Musliner, & Krebsbach 2001) must "trick" the individual agents into building collaborative plans by presenting them with controller synthesis problems that have been automatically crafted to represent their joint behavior commitments. The AMP tells the sensing agent he cannot autonomously defeat the threat, but he can communicate a warning (and that it will magically lead to the desired action) to defeat the threat. The AMP then tells the acting agent that he cannot sense, but that a warning may arrive at any time, after which he must take an action before an

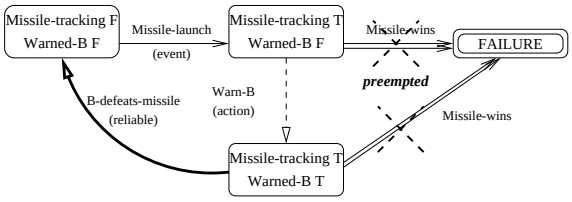

Figure 5: Agent A detects the threat and warns Agent B with a message guaranteed to be sent after no more than ΔA seconds.

```
#<Agent A TAP>
 Tests: (AND (MISSILE-TRACKING T)
             (WARNED-B F))
 Acts : WARN-B
```

Figure 6: Agent A's TAP for coordinating a preemption with Agent B. A's responsibility is to sense the condition and warn B.

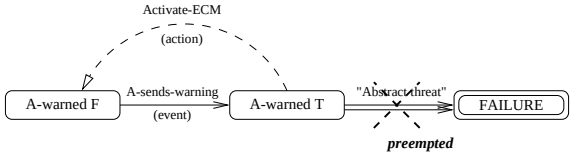

Figure 7: Agent B guarantees to detect the message from Agent A and activate its ECM within ΔB seconds, thus ensuring that the "round-trip" delay from sensing to communication to action is bounded within the maximum available time constraint.

```
#<Agent B TAP>
 Tests: (A-WARNED T)
 Acts : ACTIVATE-ECM
```

Figure 8: Agent B's TAP for listening for A's warning and taking the preemptive action in time to defeat the missile threat.

upper bound delay or face catastrophic consequences.

For a coordinated preemption, we must decompose the timing constraint imposed by a temporal transition to failure into a set of tighter constraints corresponding to bounds on the sensing, communication, and action delays of the distributed agents responding to the threat.

For example, suppose our example missile threat has a minimum delay of 8 seconds (i.e., at least 8 seconds must elapse after the missile is first tracking an agent before it can destroy the agent). This would correspond to a minimal launch and flight time considering the worst-case scenario from the agent's point of view (e.g., that the missile was launched as closely to the target agent as possible, etc.).

In a single-agent preemption, the CIRCA agent would simply have to ensure that it would detect the launch and respond with ECM activation in no more than the given 8 seconds. If the ECM device takes one second to defeat the missile tracking mechanisms, then CIRCA would recognize that it could activate the ECM as much as seven seconds after missile launch and still remain safe. So, CIRCA would build a TAP that must be polled no more than seven seconds apart.

In the coordinated preemption case, we break up the overall response time constraint (ΔT) into two parts (ΔA and ΔB) corresponding to the time that can be used by the two agents. The sensing agent (Agent A) will have to detect the threat and then send a message to the acting agent (Agent B), all within ΔA time units. Note that the communication action will mimic a regular domain action, having some associated delay (ΔA_c) and being triggered by a polling TAP just as above. Figure 5 and Figure 6 illustrate this type of plan (and corresponding TAP) for Agent A. Note that Agent A's model contains an explicit representation of Agent B's commitment to act in response to the message. The bold `B-defeats-missile` arrow represents a *reliable temporal transition*, indicating that Agent B's action places both a lower and upper delay bound on the transition's source state(s). When setting up CSM problem configurations for a coordinated preemption, the respective AMPs will generate these types of "virtual transitions" to represent the commitments of other agents.

Figure 7 and Figure 8 show that Agent B is given a representation of Agent A's possible warning of the missile threat, but no explicit representation of that threat itself. This captures the notion that Agent B cannot actually sense the threat itself, and relies on other agents for information. As with the reliable temporal transition representing Agent B's action to Agent A, here we have an *event* representing Agent A's action to Agent B. The threat of the missile is trans-

lated into a more abstract threat with a minimum delay of ΔB. Agent B must detect the warning and activate its ECM to preempt the perceived threat, and does so in the normal single-agent fashion.

Conclusion

In this paper, we have introduced the notion of *coordinated preemption*, a multi-agent extension of guaranteed failure preemption in CIRCA. Coordinated preemption allows a team of distributed CIRCA agents to build and execute synchronized plans that include joint actions such as, "You sense, I'll act". This new capability furthers our goal of extending CIRCA to multi-agent, real-time, mission-critical domains. While we have not yet implemented coordinated preemptions in CIRCA, inter-agent communication is in place for both plan-time negotiation (between different agents' AMPs and CSMs), and for run-time negotiation (between agents' RTSs).

Several research questions also remain. For example, how should the temporal delay ΔT, originally for one agent, be split into two components? How much of the delay should each agent receive, considering that these load levels influence the plan's schedulability for each agent? Because the knowledge needed to determine a feasible distribution of the available response time (if it exists) is itself distributed across agents, we will consider iterative negotiation between the coordinating agents as a first approach.

Acknowledgments

This material is based upon work supported by DARPA/ITO and the Air Force Research Laboratory under Contract No. F30602-00-C-0017.

References

Atkins, E.; Durfee, E. H.; and Shin, K. G. 1996. Plan development using local probabilistic models. In *Proc. Conf. on Uncertainty in Artificial Intelligence*, 49–56.

Goldman, R. P.; Musliner, D. J.; Krebsbach, K. D.; and Boddy, M. S. 1997. Dynamic abstraction planning. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, 680–686. Menlo Park, CA: American Association for Artificial Intelligence.

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*.

Musliner, D. J.; Krebsbach, K. D.; Pelican, M.; Goldman, R. P.; and Boddy, M. S. 1998. Issues in distributed planning for real-time control (extended abstract). In *Working Notes of the AAAI Fall Symposium on Distributed Continual Planning*.

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.