Appears in 3rd International NASA Workshop on Planning and Scheduling for Space
October 27-29, 2002; Houston, TX

Building Coordinated Real-Time Control Plans

David J. Musliner, Michael J.S. Pelican, Kurt D. Krebsbach[*]
Honeywell Technology Center
3660 Technology Drive
Minneapolis, MN 55418
{david.musliner,mike.pelican}@honeywell.com

Abstract

We are interested in developing multi-agent systems that can provide real-time performance guarantees for critical missions that require cooperation. 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 can build plans in which different agents use their unique capabilities to guarantee that the team will respond in a coordinated fashion to mission-critical events. By reasoning explicitly about different agent roles, the agents can identify what communications must be exchanged in different situations. And, by reasoning explicitly about domain deadlines and communication time, the agents can build reactive plans that provide end-to-end performance guarantees spanning multi-agent teams.

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 constellations of satellites, cooperative teams of space probes, and coordinated UAVs. In such coarse-grain distributed applications, multiple autonomous agents are each controlled by a CIRCA system, each of which builds a control plan incorporating run-time cooperation 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. In this paper, we use a simple example involving multiple spacecraft to describe CIRCA's new capabilities to negotiate coordinated roles, plan run-time communication to support coordination, and execute automatically-generated plans that ensure real-time coordination across a team of agents.

[*]Now at Lawrence University in Appleton, WI, kurt.krebsbach@lawrence.edu.

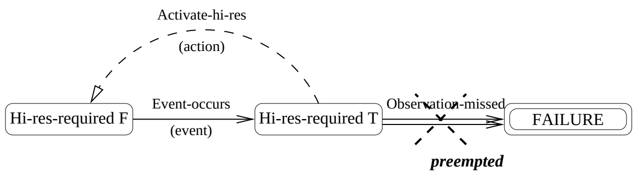

Figure 1: A simple single-agent, single-action preemption example.

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. For example, suppose a spacecraft's mission requires it to monitor for some events across a broad area and, when one of those events occurs, to focus a higher-resolution sensor on the area of interest within a short deadline (to ensure that the high-resolution sensor observes the phenomenon of interest). This situation might arise in a mission to observe geothermal activity or to identify strategic missile launches. Figure 1 shows a simple example of a state space diagram for preemption in which the agent will activate the high-resolution sensor in time to avoid missing the observation. If the system has guaranteed to detect a state in which `(Hi-res-required T)` holds, and perform a responsive action before the window for observation closes, 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).

Now suppose one satellite has a wide angle infra-red imaging sensor which identifies sites of interest that require further sensing using a high-resolution visual band sensor carried by a different satellite. The two spacecraft must coordinate their activities to preempt the missed observation failure. By *coordinated preemption*, we mean a set of complementary plans that can be executed by distributed agents to detect and react to situations before system failure occurs. How can two (or more) distributed agents build their plans to accomplish a coordinated preemption: "**You sense** the opportunity and **I'll act** on 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 hot spot fast enough, but also to tell the other agent about the opportunity quickly enough. Likewise, the acting agent must focus sufficient attention on "listening" for a message from the sensing agent at a high enough frequency that it can guarantee to both receive the message and act on the opportunity, 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 that the data collection is performed in time. In complementary fashion, the acting agent must construct a plan that honors its commitment to the acting agent. If one of the agents cannot construct a plan that satisfies its commitments, it must inform the others.

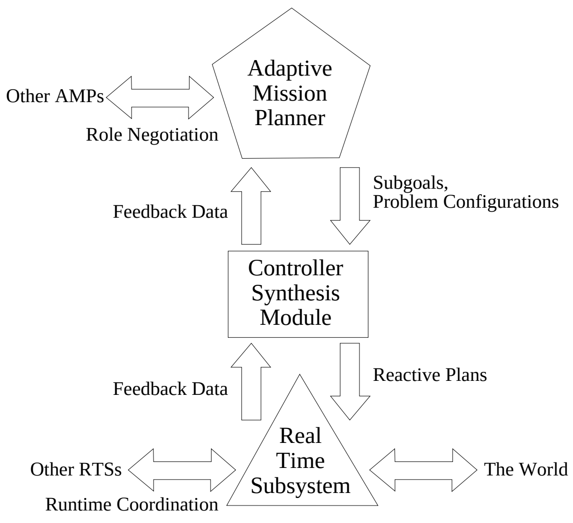

Figure 2: The CIRCA architecture combines intelligent planning and adaptation with real-time performance guarantees.

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).

All three of CIRCA's planning and execution subsystems operate in parallel. The Adaptive Mission Planner (AMP) reasons about collaborations at the level of mission roles and responsibilities using a Contract Net negotiation protocol (Musliner & Krebsbach 2001; Smith 1977). When an AMP has negotiated a responsibility for handling a particular threat or goal during a phase of a mission, it can configure a planning problem for the lower-level Controller Synthesis Module (CSM) to solve. The CSM takes in problem configurations in the form of a set of transitions (see Figure 3). The CSM reasons about an internal model of the world deduced from these transitions and dynamically programs the RTS with a planned set of reactions (Musliner, Durfee, & Shin 1995; Goldman *et al.* 1997). While the RTS is executing those reactions, ensuring that the system avoids failure, the AMP and CSM 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 events, maintaining system safety. When the new reaction set has been developed, it can be downloaded to the RTS.

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 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.

Figure 3 contains three very simple transitions from a single-agent version of our example observation satellite domain, corresponding to the planned state space diagram in Figure 1. The `event-occurs` event transition represents the occurrence of the phenomenon of interest, out of the agent's control. That event sets the preconditions for the `observation-missed` temporal transition, which may occur as little as 500 milliseconds after the preconditions are established. Because the postconditions of `observation-missed` include `(failure T)`, it is a **temporal transition to failure** (TTF). To preempt that failure, the CIRCA agent will plan to take the `activate-hi-res` action in the intervening state.

CIRCA builds its reactive plans in the form of Test-Action Pairs (TAPs) that test for a particular situation and invoke a planned response. Figure 4 shows

```
;;; The initial sensed event can occur
;;; at any time.
(def-event 'event-occurs
     :preconds '((hi-res-required F))
     :postconds '((hi-res-required T)))

;;; If you dont do hi-res sensing by 500
;;; milliseconds, you fail.
(def-temporal 'observation-missed
     :preconds '((hi-res-required T))
     :postconds '((failure T))
     :min-delay 500)

;;; In the simple single-agent case, doing
;;; this handles the event.
(def-action activate-hi-res
     :preconds '()
     :postconds '((hi-res-required F))
     :wcet 300)
```

Figure 3: CIRCA domains capture agent capabilities and world dynamics as transitions.

```
#<TAP 1>
 Tests: (HI-RES-REQUIRED T)
 Acts : ACTIVATE-HI-RES
```

Figure 4: Single agent Test-Action Pair to activate a high-resolution sensor when an observation is required.

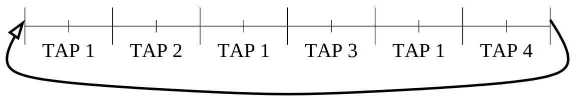

Figure 5: A TAP schedule loop in which TAP 1 must be run more often than the others.

the TAP automatically generated and scheduled by CIRCA to implement the simple preemption in our running single-agent example. 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 CSM 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 CSM 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 5 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.

Negotiating Coordinated Roles

How do the agents decide which role they are playing, and what to communicate about? In our current implementation, each CIRCA agent has a representation of its own capabilities, expressed as the classes

```
(def-event 'event-occurs
     :preconds '((saw-event F))
     :postconds '((saw-event T)))

;;; If you don't do hi-res sensing by
;;; 500 msec, fail.
(def-temporal 'hi-res-observation-missed
     :preconds '((saw-event T))
     :postconds '((failure T))
     :min-delay 500)

;;; All the sensor agent can do is notify
;;; the hi-res agent.
(def-action 'notify-hi-res
     :preconds '((notified-hi-res F))
     :postconds '((notified-hi-res T))
     :wcet 10)

;;; This reliable temporal represents
;;; sensor agent's model of actor agent's
;;; commitment to respond.
(def-reliable 'hi-res-observes
     :preconds '((saw-event T)
                 (notified-hi-res T))
     :postconds '((saw-event F)
                  (notified-hi-res F))
     :delay (make-range 250 400))
```

Figure 6: The sensor agent can detect the impending failure, but cannot directly prevent it.

```
;;; The uncontrollable event can occur
;;; at any time.
(def-event 'hear-about-event
     :preconds '((heard-about-event F))
     :postconds '((heard-about-event T)))

(def-action 'simple-observe-event
     :preconds '((heard-about-event T))
     :postconds '((heard-about-event F))
     :wcet 300)

;;; If you don't observe event in hi-res by
;;; 400 ms after notification, you've failed.
(def-temporal 'observation-missed
     :preconds '((heard-about-event T))
     :postconds '((failure T))
     :min-delay 400)
```

Figure 7: The simplest actor model gets a message and must respond.

of goals and threats (potential failures) it can handle. When a new mission is presented to a multi-CIRCA system, the agents each bid on the component goals and threats that make up the mission, based on their capabilities.

In our running example, the mission is characterized by two distinct threats, one representing the sensor agent's need to send a message to the actor agent by a deadline, and one representing the actor agent's need to respond to the message by a deadline. The respective agents bid to handle these threats, and win their appropriate roles on the team. It is worth noting that this decomposition is already present in the mission description entered by the system programmer or tasking agent; future versions may be able to decide themselves whether to tackle the response to a threat in a centralized or cooperative distributed fashion.

To get the CSMs to plan the coordination communication explicitly, the Adaptive Mission Planner (AMP) 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 sensing agent's AMP tells its CSM that it cannot autonomously defeat the threat, but that if it can communicate a warning quickly enough, this warning will lead to the defeat of the threat (see Figure 6). The actor agent's AMP tells its CSM that it cannot sense the threat directly, but that a warning may arrive at any time, after which it must take an action before an upper bound delay or face catastrophic consequences (see Figure 7).

Building Coordinated Plans

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 threat (a critical high-resolution observation) has a minimum delay of 500 milliseconds (i.e., at least 500 milliseconds must elapse after the appropriate surface event has occurred, before the phenomenon disappears or expires, causing the team to have missed an observation). This would correspond to the minimum expected phenomenon duration, and hence the worst case that makes it hardest to observe.

In a single-agent preemption, the CIRCA agent would simply have to ensure that it would detect the event and respond with hi-res sensor activation in no more than the given 500 milliseconds. If the hi-res sensor takes no more than 300 milliseconds to capture its observation, then CIRCA would recognize that it could activate the hi-res sensor as much as 200 milliseconds after the event and still remain safe. So, CIRCA would build a TAP that must be polled no more than 200 milliseconds 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 8 and Figure 9 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 mes-

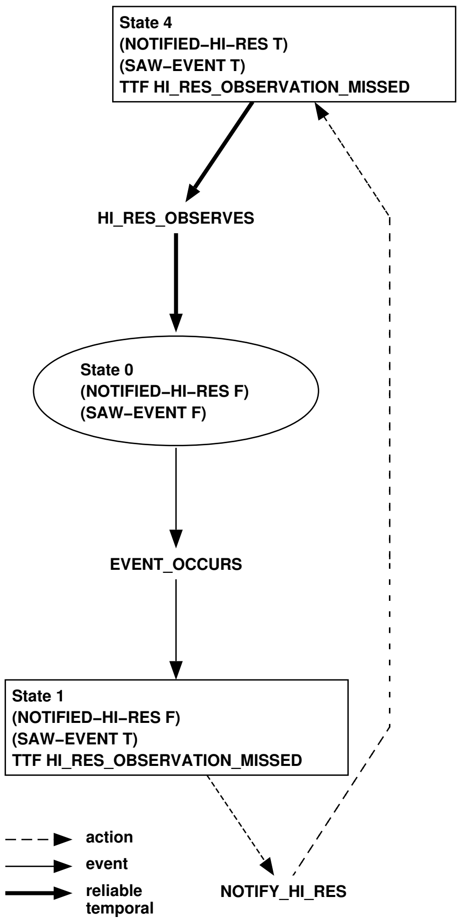

Figure 8: Agent A detects the threat and warns Agent B with a message guaranteed to be sent after no more than ΔA seconds.

```
#<Agent A TAP>
 Tests: (AND (SAW-EVENT T)
             (NOTIFIED-HI-RES F))
 Acts : NOTIFY-HI-RES
```

Figure 9: Agent A's TAP for coordinating a preemption with Agent B. A's responsibility is to sense the condition and notify B.

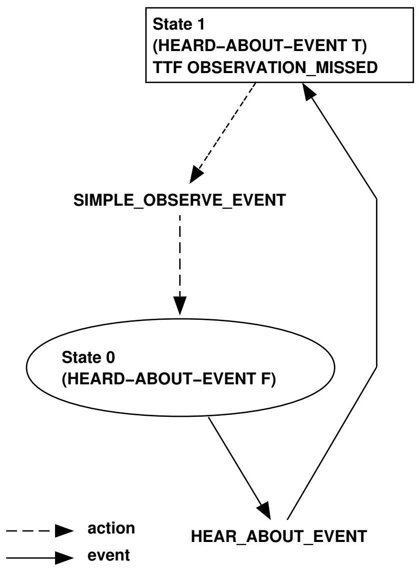

Figure 10: Agent B guarantees to detect the message from Agent A and activate its hi-res sensor within ΔB seconds, thus ensuring that the "round-trip" delay from sensing to communication to action is bounded within the maximum available time constraint.

```
#<Agent B TAP>
 Tests: (HEARD-ABOUT-EVENT T)
 Acts : SIMPLE-OBSERVE-EVENT
```

Figure 11: Agent B's TAP for listening for A's warning and taking the preemptive action in time to avoid mission failure.

sage. The bold `hi-res-observes` 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 include these types of "virtual transitions" to represent the commitments of other agents.

Figure 10 and Figure 11 show that Agent B is given a representation of Agent A's possible notification of the event, but no explicit representation of that event itself. This captures the notion that Agent B cannot actually sense the threat directly, 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 (sending a message) to Agent B. The threat of the impending observation deadline is translated into a more abstract threat with a minimum delay of ΔB. Agent B must detect the warning and activate its hi-res sensor to preempt the perceived deadline, and does so in the normal single-agent fashion.

Of course, this trivial example makes the problem look simple. The challenge arises when the sensing, communicating, and acting responsibilities are more complicated, and must be intertwined with other agent activities. To give an idea of the complexity that rapidly sets in, consider an example only slightly more complicated. Suppose that the agent with the hi-res sensor must actually activate a fine-grain alignment system to point the sensor, and only when that alignment system is maintaining tight alignment can the hi-res sensor be used. In that case, the domain description is not much larger (see Figure 12), but the state space that results becomes considerably more complicated (see Figure 13).

Handling this complexity growth is an ongoing research challenge; promising approaches include using automatic, dynamic abstraction to omit state information when feasible (Goldman *et al.* 1997) and using decision-theoretic methods to trade off completeness for computability (Goldman, Musliner, & Krebsbach 2001).

Related Work

Other work on multi-agent team coordination has focussed on "joint intentions" and using explicit models of team plans to define individual agent plans (Tambe 1997), often using constraint satisfaction and on-line repairs to adapt to evolving environments and conflicts between agent plans (Jung, Tambe, & Kulkarni 2001; Tate, Levine, & Dalton 1999). While the higher levels of the CIRCA architecture do support elements of this approach, our current work is focused on building plans that accurately model and control the dynamics of coordination between agents at run-time. In particular we are interested in real-time, dependable interactions between teammates.

For the most part, other planning and execution systems for spacecraft handle multi-agent coordination in a more mission-specific fashion. For example, the automated mission planner for the Modified Antarctic Mapping Mission (MAMM) coordinates downlinks from RADARSAT to ground stations (Smith, Englehardt, & Mutz 2002). However, the mission planner does not plan ground station activities except as implied by the spacecraft plan. And although the MAMM's domain is certainly dynamic, the mission planner treats it as predictable. The mission plan is fixed. If an observation is missed, the remaining mission must be replanned.

CASPER, like CIRCA, provides a soft real-time onboard planning facility (Chien *et al.* 1999). Unlike CIRCA, CASPER builds plans for a sequential task executive, and repairs them when necessary to react to environment dynamics. CIRCA's TAP schedules incorporate real-time reactions to an unpredictable environment; environmental dynamics can be handled in the executive directly. Dynamics that exceed the scope of the pre-planned reactions are handled by on-the-fly replanning at the CSM and AMP levels.

All of the ASPEN family of planners, including CASPER and the MAMM mission planner, include specific resource models and constraint checking (Chien *et al.* 2000). CIRCA represents resources implicitly, as postcondition or precondition features in the CSM and as roles or capabilities in the AMP. Plans for future work include explicit resource representation and checking.

Both the Three Corner Sat Mission and the Autonomous Sciencecraft Constellation projects are extending ASPEN-based planners to handle distributed teams of spacecraft like those described in our examples (Chien *et al.* 2001a; 2001b).

Conclusion

In this paper, we have discussed 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. We have implemented coordinated preemptions in CIRCA, using inter-agent communication for both plan-time negotiation (between different agents' AMPs), and for run-

```
(def-action 'align-hi-res-sensor
        :preconds '()
        :postconds '((hi-res-sensor-aligned T)
                    (hi-res-sensor-effective T))
        :wcet 10)

;; Once you align sensor, can trigger it and take detailed reading.
(def-action 'begin-hi-res-sensing
        :preconds '((sensing normal)(hi-res-sensor-aligned T)
                     (hi-res-sensor-effective T))
        :postconds '((sensing hi-res))
        :wcet 10)

;; After a while, the sensor alignment expires...
(def-temporal 'sensor-alignment-expires
        :preconds '((hi-res-sensor-aligned T))
        :postconds '((hi-res-sensor-aligned F))
        :min-delay 100)

;; After sensor alignment expires, a while later the effectiveness is gone;
;; we should preempt this transition to keep the sensing reliable temporal working.
(def-temporal 'sensor-ineffective
                    :preconds '((hi-res-sensor-aligned F)(hi-res-sensor-effective T))
                    :postconds '((hi-res-sensor-effective F))
                    :min-delay 100)


;; need hi-res-sensor-effective, or the sensing doesn't work...
(def-reliable 'observe-event
        :preconds '((heard-about-event T) (sensing hi-res)
                        (hi-res-sensor-effective T))
        :postconds '((heard-about-event F))
        :delay (make-range 250 300))

(def-action 'end-hi-res-sensing
        :preconds '((sensing hi-res))
        :postconds '((sensing normal))
        :wcet 10)
```

Figure 12: A few additional transitions can define a much more complex plan space.

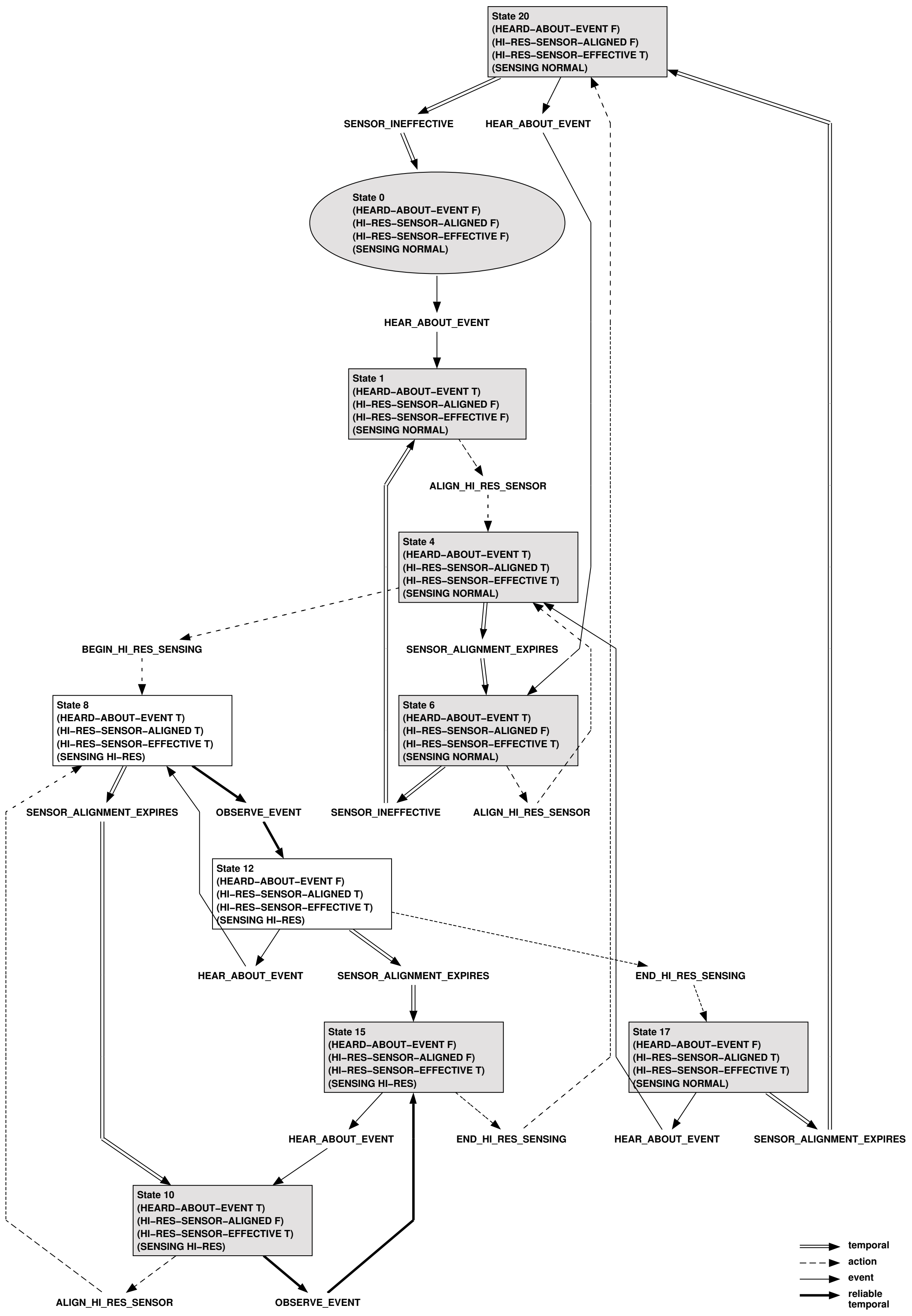

Figure 13: When Agent B must maintain sensor alignment, the reachable state space grows quickly.

time coordination (between agents' RTSs).

Several research questions also remain. For example, how should the available response delay ΔT, originally for one agent, be split into two or more 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

Chien, S.; Knight, R.; Stechert, A.; Sherwood, R.; and Rabideau, G. 1999. Using iterative repair to increase the responsiveness of planning and scheduling for autonomous spacecraft. In *IJCAI99 Workshop on Scheduling and Planning meet Real-time Monitoring in a Dynamic and Uncertain World.*

Chien, S.; Rabideau, G.; Knight, R.; Sherwood, R.; Engelhardt, B.; Mutz, D.; Estlin, T.; Smith, B.; Fisher, F.; Barrett, T.; Stebbins, G.; and Tran, D. 2000. ASPEN – Automating space mission operations using automated planning and scheduling. In *SpaceOps 2000.*

Chien, S.; Engelhardt, B.; Knight, R.; Rabideau, G.; Sherwood, R.; Hansen, E.; Ortiviz, A.; Wilklow, C.; and Wichman, S. 2001a. Onboard autonomy on the Three Corner Sat mission. In *Proceedings of ISAIRAS 2001.*

Chien, S.; Sherwood, R.; Burl, M.; Knight, R.; Rabideau, G.; Engelhardt, B.; Davies, A.; Zetocha, P.; Wainright, R.; Klupar, P.; Cappelaere, P.; Surka, D.; Williams, B.; Greeley, R.; Baker, V.; and Doan, J. 2001b. The Techsat-21 autonomous sciencecraft constellation demonstration. In *Proceedings of ISAIRAS 2001.*

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 Int'l Workshop on Self Adaptive Software.*

Jung, H.; Tambe, M.; and Kulkarni, S. 2001. Argumentation as distributed constraint satisfaction: applications and results. In *Proc. of the Fifth Int'l Conf. on Autonomous Agents*, 324–331.

Musliner, D. J., and Krebsbach, K. D. 2001. Multi-agent mission coordination via negotiation. In *Working Notes of the AAAI Fall Symposium on Negotiation Methods for Autonomous Cooperative Systems.*

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 Transactions on 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.

Smith, B. D.; Englehardt, B. E.; and Mutz, D. H. 2002. The RADARSAT-MAMM automated mission planner. *AI Magazine.*

Smith, R. 1977. The contract net: A formalism for the control of distributed p roblem solving. In *Proc. Int'l Joint Conf. on Artificial Intelligence*, volume 1, 472.

Tambe, M. 1997. Implementing agent teams in dynamic multi-agent environments. *Applied Artificial Intelligence.*

Tate, A.; Levine, J.; and Dalton, J. 1999. Using ai planning techniques for army small unit operations. In *Proc. of the Fifth Int'l Conf. on Artificial Intelligence Planning and Scheduling Systems (AIPS 2000).*