CSC: Criticality-Sensitive Coordination *

Pedro Szekely†, Marcel Becker‡, Stephen Fitzpatrick‡, Gergely Gati⋆,
David Hanak⋆, Jing Jin†, Gabor Karsai⋆, Rajiv T. Maheswaran†,
Bob Neches†, Craig M. Rogers†, Romeo Sanchez†, Chris van Buskirk⋆

† Information Sciences Institute, University of Southern California, 4676 Admiralty Way - Suite 1001, Marina Del Rey, CA 90292

‡ Kestrel Institute, 3260 Hillview Avenue, Palo Alto, CA 94304

⋆ Institute for Software Integrated Systems, Vanderbilt University, Box 1829, Station B, Nashville, TN 37235

1. Introduction

Our Criticality-Sensitive Coordination (CSC) agents are designed to enhance the performance of a human-team working together in uncertain and dynamic settings by monitoring and adapting their plans as dictated by the evolution of the environment. Such situations model military scenarios such as a coordinated joint operations or enterprise settings such as multiple-project management. Among the many challenges in these situations are the large space of possible states due to uncertainty, the distributed / partial knowledge of current state and plan among the agents and the need to react in a timely manner to events that may not be in the original model. In fact, reaction alone is often insufficient as in environments where success depends on completing sequences of coupled actions, one needs to anticipate future difficulties and enable contingencies to alleviate potential hazards.

2. System Description

To deal with these challenges, the CSC agents are constructed with components that interact in a multi-tiered manner with functionalities that operate in various timescales and reasoning domains. The system architecture is depicted in Figure 1. The highest level of reasoning is performed by three components: (1) the deliberative scheduler, (2) the opportunistic scheduler, and (3) the downgrader. The deliberative scheduler is triggered when it has been determined that a portion of the existing plan is anticipated to fall below a certain likelihood of success. It then performs *dynamic partial-centralization* of the relevant agents and nodes in the plan, extracts potential solutions and proposes schedule modifications to remedy the problem. Due to the complexity of this task and the nature of the anticipated failure, this process occurs over several decision epochs with decision windows that begins slightly in the future. The opportunistic scheduler performs schedule modifications in the near-term. Using only local information, it is capable of making decisions on a faster time-scale, and buffers the existing plan by utilizing free resources. This provides robustness by increasing the probability of success and generating opportunities to obtain higher quality outcomes. The downgrader complements the previous two components by freeing resources both in the near-term and the future based on the evolution of the system up to the present. The schedule, estimates of system state and metrics of uncertainty are kept in the state manager which communicates with the state managers of other agents to distribute and collect locally visible and locally relevant information. The *profiles* which capture both system state estimates, metrics of uncertainty and potential solutions to problems are propagated in a decentralized manner on the order of a single decision epoch. Each agent has profiles and schedules both for itself, for remote agents that affect it directly and for plan components for which it is responsible. The local schedule information is sent to an execution controller which interacts with the environment. This component is isolated such that it can operate at a much faster time-scale than the decision epoch intervals, such that the agent can continue to function even under high computational burdens in other components of the system.

3. Demo Description

The capabilities of the system and the complexities of the problem are illustrated in a scenario where two subteams, represented by CSC agents Alpha and Bravo, participate in a three-phase joint project/operation. Intiially, they

* The work presented here is funded by the DARPA COORDINATORS Program under contract FA8750-05-C-0032. The U.S.Government is authorized to reproduce and distribute reports for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of any of the above organizations or any person connected with them.

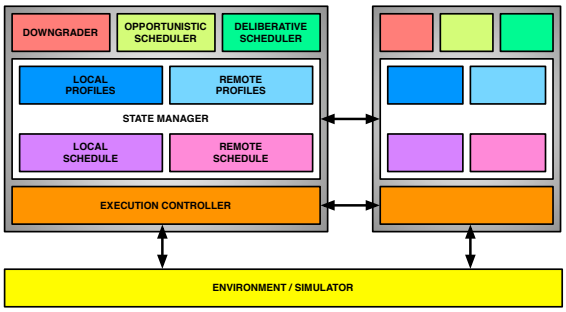

Figure 1. CSC Agent Architecture

agree to engage in Plan A, which involve each agent's subteam performing certain activities at each phase. However, they can fall back to Plan B, which involves different activities at each phase, if needed. The activities are not independent as one may need to perform a certain activity in an earlier phase to have the option of performing an activity in a later phase. Thus, the necessity of anticipation in the presence of uncertainty. A depiction of an evolution is shown and discussed in Figure 2.

Without CSC, the initial plan can be undermined in many ways. A delay of an activity in Phase 1 cascades the delay on to Phase 2 implying a high chance of failure. While Phase 2 might succeed despite the failure of a single activity within it, Phase 3 will be completely damaged due to dependencies of activities across multiple phases. However, the various components of CSC interact to allow adverse circumstances to be ameliorated with ease. The state manager for the agent will detect in Phase 1 itself that the probability of success for activities in Phase 3 have fallen below critical levels and will instantiate dynamic partial-centralization by calling the deliberative scheduler. The deliberative scheduler will then consider alternatives, in this case Plan B, and install new activities in the second phase and third phase. The downgrader frees up the resources in Phase 2 that were operating under the directives of Plan A, such that they are available for Plan B to execute. Finally, the opportunistic scheduler utilizes resources that were freed up in Phase 3 (by the downgrader) to add an activity that enhances the quality of the solution.

4. Starfields

One of the unique and extremely beneficial aspects of CSC is the vast suite of tools (referred to as *starfields*) available for visualizing various components and the evolution of the system. These tools offer the ability to observe behavior at a macro-level. This helps a user quickly and accurately gauge system behavior and also provides the ability to isolate individual components at particular instants in time which is an invaluable aid in debugging. A snapshot of selected starfields is displayed in Figure 3. A key functionality of starfields is the ability to play back any evolution in time to observe the system at chosen instants. This allows a user to identify exactly when and how the system identifies and resolves problems. Another key feature is the ability to search and cross-reference activities and plan components, which offers the capability to isolate an activity in multiple starfields and see how it couples with activities in other phases of the plan. The starfields facilitated rapid modification, development and debugging in addition to aiding in the understanding of the underlying problem which becomes difficult to grasp as the scale becomes large. A movie of the demonstration can be downloaded from *http://www.isi.edu/~szekely/csc/aamas06/csc-demo-v01.html*.

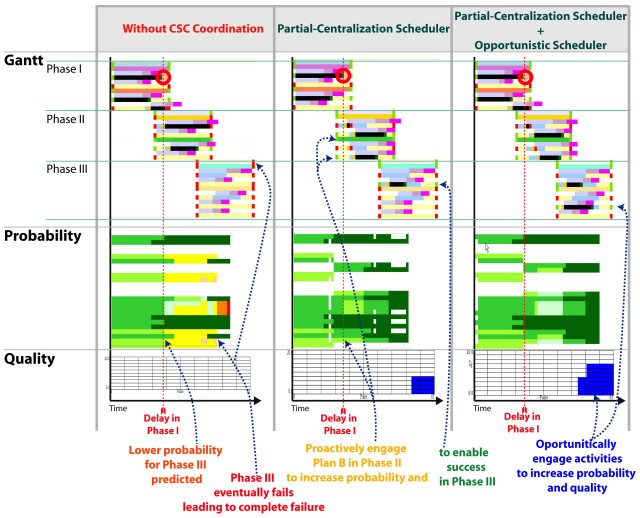

Figure 2. Two-Agent Three-Phase Example

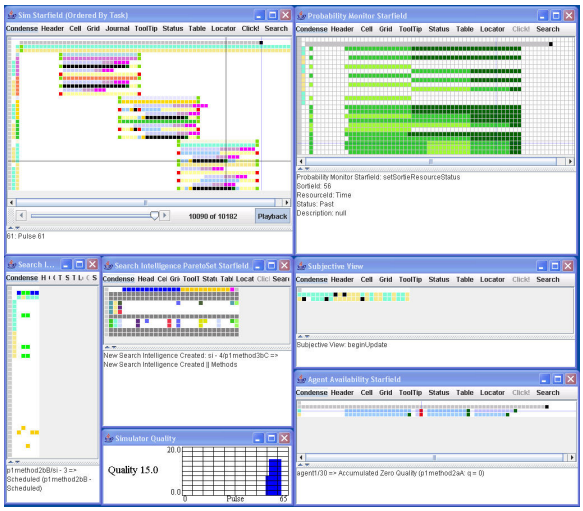

Figure 3. CSC Starfields