ARTICLE IN PRESS

Available online at www.sciencedirect.com

ScienceDirect

Engineering Applications of Artificial Intelligence ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Engineering Applications of ARTIFICIAL INTELLIGENCE

www.elsevier.com/locate/engappai

Viewpoint

Airlift mission monitoring and dynamic rescheduling

David E. Wilkins[a], Stephen F. Smith[b], Laurence A. Kramer[b], Thomas J. Lee[a], Timothy W. Rauenbusch[a,*]

[a]*Artificial Intelligence Center, SRI International, Menlo Park, CA 94025, USA*
[b]*The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA*

Received 10 October 2006; accepted 2 April 2007

Abstract

We describe the Flight Manager Assistant (FMA), a prototype system, designed to support real-time management of airlift operations at the USAF Air Mobility Command (AMC). In current practice, AMC flight managers are assigned to manage individual air missions. They tend to be overburdened with associated data monitoring and constraint checking, and generally react to detected problems in a local, myopic fashion. Consequently, decisions taken for one mission can often have deleterious effects on others. FMA combines two key capabilities for overcoming these problems: (1) intelligent monitoring of incoming information (for example, weather, airport operations, aircraft status) and recognition of those situations that require corrective action and (2) dynamic rescheduling of missions in response to detected problems, both to understand the global implications of changed circumstances and to determine appropriate rescheduling actions. FMA builds on two existing technologies: an execution-monitoring framework previously applied to small-unit operations and control of robots, and a dynamic scheduling tool that is transitioning into operational use in AMC's Tanker/Airlift Control Center. FMA's dynamic-mediation module provides for collaborative mission management by different planning and execution offices by structuring communication for decision making.

Keywords: Execution monitoring; Scheduling; Dynamic schedule repair; Collaboration; Mediation

1. Introduction and problem statement

Management of flight operations at the United States Air Force Air Mobility Command (AMC) is a challenging problem. AMC typically flies several thousand missions worldwide on a weekly basis (more in a crisis situation), involving several hundreds of aircraft and comparable numbers of aircrews. The execution of any given mission requires attention to a broad range of constraints relating to the mission's requirements (e.g., delivery dates, cargo type and weight), resource availability (e.g., aircraft, aircrews, airports, diplomatic clearances) and usage constraints (e.g., crew duty-day restrictions and scheduled return dates, aircraft speed, range, and capacity, airspace restrictions). Although missions are planned and globally scheduled to satisfy such constraints, the dynamics of execution regularly forces changes. Aircraft break down, airports become unavailable due to weather, missions become delayed due to diplomatic clearance problems, and so on, and all such events can warrant reassessment of previous allocation decisions. In such execution-driven rescheduling contexts, it is important to weigh potential recovery options against their prospective impact on future operations, and to take actions that continue to make the most effective global use of AMC assets.

In current practice, management of flight operations at AMC is a stove-piped process, where planning and execution are treated as sequential steps and information flows in one direction (from planning to execution). New mission requirements flow into AMC's planning offices on a continuous basis, and as they do, aircraft and aircrews are incrementally allocated to support new missions in accordance with associated priorities and as resource

*Corresponding author.
E-mail addresses: wilkins@ai.sri.com (D.E. Wilkins), sfs@cs.cmu.edu (S.F. Smith), lkramer@cs.cmu.edu (L.A. Kramer), tomlee@ai.sri.com (T.J. Lee), rauenbusch@ai.sri.com (T.W. Rauenbusch).

0952-1976/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.engappai.2007.04.001

availability allows. When a mission gets to within 24 h of execution, it is "pushed" from the planning side of AMC to the execution office, and becomes the responsibility of an individual flight manager.

AMC flight managers take responsibility for checking to ensure that all mission constraints remain satisfied before and during execution, and as problems are detected, they diagnose and revise mission plans to facilitate mission continuation and/or recovery. In practice AMC flight managers are not as well supported in this execution-management task as they could be. Some alerting tools do exist for signaling certain kinds of problems, but there is generally no ability to differentiate routine checks from exceptional events (i.e., everything shows up red), and no ability to detect more complex, compound conditions. Flight managers are typically overburdened by the data monitoring and constraint checking that are required to ensure the continuing viability of executing missions. Furthermore, when problematic situations are detected, flight managers have no visibility of the larger AMC operating picture, and must take recovery actions without regard to potential interactions with other missions. As a result, execution management often proceeds in fire-fighting mode, where putting out one fire ignites the next one.

For the past several years, we have been developing technologies that provide a basis for more effective flight management within AMC. We briefly describe the key technologies.

- At Carnegie Mellon University (CMU) we have been developing the AMC Allocator, a dynamic scheduling tool for day-to-day management of airlift and air refueling schedules (Smith et al., 2004; Kramer and Smith, 2002). The AMC Allocator provides a range of capabilities for incrementally revising schedules to accommodate new or changed requirements, with continued emphasis on efficient resource utilization. It is currently transitioning into use as a *planning* tool in the Tanker/Airlift Control center at AMC.
- At SRI International, we have been developing an execution-monitoring framework that encodes and applies knowledge-based, task-specific, monitoring techniques, and uses the concept of *value of an alert* to control interaction with humans. The first application of this framework was the Small Unit Operations Execution Assistant (SUO-EA), which monitors large volumes of situational data and gets urgent, plan-aware alerts to the right users. SUO-EA was successfully demonstrated in the DARPA SUO program, and the framework has since been applied to other problems (Wilkins et al., 2003).
- SRI has also developed technologies for incremental negotiation and coalition formation within the DARPA Autonomous Negotiating Teams program and the ONR UCAV program (Ortiz et al., 2003).
- SRI's Open Agent Architecture (OAA) (Cheyer and Martin, 2001) provides a robust integration infrastructure that has been used in dozens of applications.

In this paper, we describe the Flight Manager Assistant (FMA), a system developed under the Air Force Research Laboratory's Integrated Flight Management (IFM) advanced technology demonstration program. FMA integrates the above set of technology components to provide a flexible, mixed-initiative tool for real-time flight management. Through a coupling of an execution-monitoring framework with a global dynamic scheduler (DS), the FMA is designed to promote a more integrated, and hence more informed basis for detecting and responding to exceptional execution events.

The FMA actively monitors data-information sources for expectations it derives from the current schedule, recognizes deviations immediately, and applies policies for responding to deviations. Responses to significant deviations may alert the user to take control. Other options might include automated responses (when permitted by policy), or invoking the scheduler to explore alternative rescheduling options. By integrating status update information with the current schedule, the FMA indicates the important consequences of detected events on current and future operations.

By generating and comparing alternative schedule repairs (either interactively with the user or automatically), the FMA supports determination of globally coherent recovery actions, while promoting schedule changes that minimize disruption to other missions whenever possible. A given schedule-repair process may also initiate and assist a collaboration between the user responsible for execution and the users who planned the missions. Finally, the FMA can provide automated support for implementing the human-selected response. The FMA continuously reacts to new information while interspersing its proactive pursuit of response procedures.

The broad goal of the FMA project has been to develop technology that enables increased organizational responsiveness and effectiveness in managing the dynamics of mission operations. In our view, there are two key factors to realizing this goal:

- *Increased automation*: Ubiquitous computers, data sources, and reliable, high-bandwidth communication networks are providing too much information for humans to monitor. In our vision, flight managers will rely on an automated execution aid to monitor the large (and ever increasing) volume of incoming information. By understanding the plan and situation, such an execution aid will consider the outputs of multiple monitoring techniques and tools, and then judge when the user should be alerted. Good judgment avoids overalerting. There may be many plan deviations (called *exceptions* at AMC) noted in the current plan by various AMC monitoring tools—the FMA recognizes which are most important, focuses the human on those, and assists with developing responses.
- *Closing the loop between planning and execution*: The ability to effectively respond to important alerts requires

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

access to the global state of current and planned future operations, and to the rationale that underlies current mission plans/schedules. In our vision, flight managers will utilize dynamic scheduling tools to understand the consequences of detected events, to generate alternative reactions and evaluate the impact of each, and to provide a basis for negotiating mission requirements—the FMA provides these sorts of capabilities and enables a flight manager to apply a more global perspective in determining how to respond. The FMA also alerts originating planners to problems with their missions and provides support for them to contribute information relevant to execution decisions and achieve globally beneficial changes to individual mission plans.

The current FMA prototype is composed of two principal components: a Flight Manager Executive (built using SRI's monitoring framework) and a Dynamic Scheduler (DS) (derived from CMU's AMC Allocator system). We have demonstrated this prototype on a series of execution management vignettes, using actual (full scale) AMC schedules pulled from AMC's Consolidated Air Mobility Planning System (CAMPS), and representative (but scripted) execution data streams. A third Dynamic Mediation component (based on SRI's incremental negotiation techniques) has undergone preliminary proof-of-concept testing.

In the following sections, we describe these components in more detail, describe how the above challenges are addressed, and give an indication of the application's status and potential for transition.

2. FMA architecture

The FMA architecture features actors—software agents that model the participants in the decision-making process. The FMA is configurable for arbitrary sets of decision makers. A typical configuration includes at least one actor for the execution office and for each planning office.

Planning offices at AMC include those for Special Assignment Airlift Missions (SAAMs), Channel (periodic logistical) Missions, and Contingency Missions, Exercises and Training Missions. Planning offices include those for Special Assignment Airlift Missions (SAAMs), Channel (periodic logistical) Missions, Contingency Missions, and Exercises and Training Missions. To better respond to situations that arise during mission execution, planners have a need to convey information, such as the mission constraints or the plan rationale, to the execution office. The executive office can make better decisions by consulting with a particular planning office when it is necessary to repair execution-time problems with missions from that office. The FMA actor architecture is designed to facilitate this interaction.

The inputs that the FMA monitors come from various sources, primarily AMC software tools and messages from other actors and external agents. For example, one tool, HISA (Human Interaction with Software Agents) (Mulvehill and Whitaker, 2002), detects and reports maximum-on-ground (MOG) aircraft capacity conflicts at airbases. Another tool is IMT (Integrated Management Tool), which the Flight Managers use to monitor mission status.

The actor-based architecture facilitates a number of capabilities important to improved flight management. By using software agents in tandem with, and specific to, their human counterparts, information flow and communication can be automated and regulated according to the overall global situation, and to each human's preferences, work style, level of experience, and current workload. For example, low-priority notifications can be filtered out of the communication stream for a planner addressing a crisis situation. By keeping a store of relevant data with each actor, information is well organized for the purposes of decision making, and communication among actors can be minimized, enhancing collaboration.

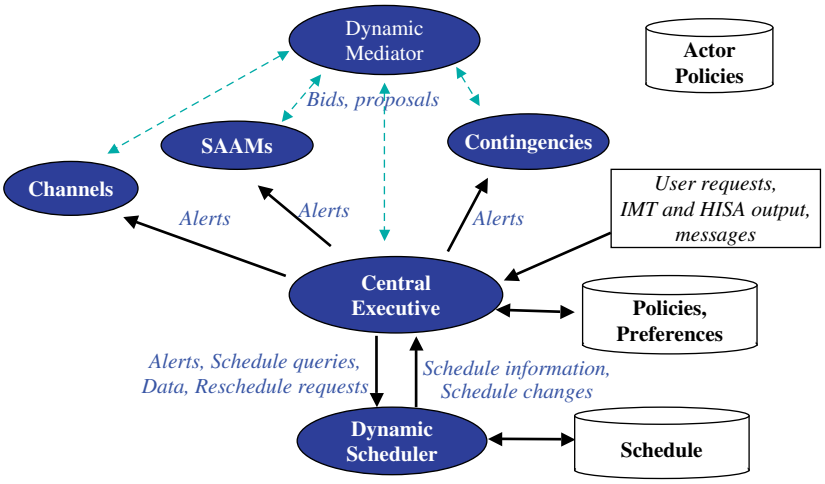

Fig. 1. FMA Architecture. The arrows represent message and information flow; every agent uses the Actor Policies (arrows omitted). The FMA monitors the output of AMC software tools such as IMT and HISA.

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

Fig. 1 depicts the various actors in a common configuration of the FMA. OAA is used to communicate between the software agents in the architecture, thus permitting the agents to be widely distributed geographically. The FMA is based on four software modules, which are described in more detail below:

- GUI
- Executive (Exception Handler)
- Dynamic Scheduler (DS)
- Dynamic Mediator (DM)

2.1. Executive

The key problem for the Executive is that algorithms that alert on constraint violations and threats in a straightforward manner inundate the user with alerts. Unwanted alerts are a problem in many domains, from medicine to transportation to battle command. For example, the problem of flooding human users with false or redundant alarms is ubiquitous in medical monitoring (Tsien, 1997). One study found that 86% of alarms in a pediatric ICU were false alarms (Tsien and Fackler, 1997). An execution aid that gives alerts every few seconds will quickly (if not immediately) be discarded by the user in stressful situations.

To be useful, an execution aid must produce high-value, user-appropriate alerts. Addressing this challenge has several aspects. Reactivity is important. Resources are not generally available to perform all desired analyses for every input. For example, projecting future problems with multiple simulation runs cannot generally be done for every status update. There are often no obvious boundaries to the types of support an execution aid might provide in a real-world domain. Therefore, a balance must be struck between the capabilities provided and resources used.

Another aspect of this challenge is avoiding cascading alerts as events get progressively further away from expectations along any of a number of dimensions (such as time, space, and resource availability). An aspect that we will not discuss in depth is aggregating lower-level data, which can reduce the number of alerts by consolidating inputs. Finally, alerts and their presentation may have to be adjusted to the situation, including the user's cognitive state (or the computational state of a software agent). For example, in high-stress situations, tolerances could be increased or certain types of alerts might be ignored or postponed.

2.1.1. Solving monitoring challenges

Our approach is grounded in the concept of determining the value of an alert (VOA). First, the system must estimate the value of new information to the user. We use the term value of information (VOI) to refer to the pragmatic import the information has relative to its receiver. We assume that the practical VOI derives from its usefulness in making informed decisions. This definition of VOI is different from the notion of VOI used in information theory (see Weinberger, 2002 for a discussion).

However, alerting the user to all valuable information could have a negative impact in certain situations, such as when the alert distracts the user from more important tasks, or when too many alerts overwhelm the user. We therefore introduce the additional concept of VOA, which is the pragmatic import (for making informed decisions) of taking an action to focus the user's attention on a piece of information. VOA takes VOI into account but weighs it against the costs and benefits of interrupting the user. If the user is busy doing something significantly more important, then issuing an alert might not be valuable, even when VOI is high.

Our monitoring framework integrates many domain-specific and task-specific monitoring techniques and then uses the concept of VOA to avoid operator overload. We have used this framework to implement Execution Assistants (EAs) in three different dynamic, data-rich, real-world domains to assist a human in monitoring team behavior (Wilkins et al., 2003). One domain (Army small-unit operations) has hundreds of mobile, geographically distributed agents, a combination of humans, robots, and vehicles. The second domain (teams of unmanned ground and air vehicles) has a handful of cooperating robots. Both domains involve unpredictable adversaries in the vicinity. The application to integrated flight management at AMC represents our third application.

We developed algorithms that heuristically estimate VOI (either quantitatively or qualitatively) using domain knowledge, which we obtained by interviewing subject-matter experts. These domain-specific algorithms are, and must be, easily customized and tuned for user preferences, as well as the situation. They are invoked in domain-independent ways for a variety of purposes by the monitoring framework, and were developed with feedback from domain experts. We believe it is feasible to use machine-learning techniques to replace or supplement hand-coded heuristics for VOI/VOA estimation and/or the user preferences that affect it, but this has not yet been explored.

VOIs and VOAs from various sources are combined qualitatively in our domains, using several domain-specific quantitative measures in the qualitative reasoning process. Issuing an alert is a discrete event, and generally there are a small number of options for presenting an alert. Therefore, estimating VOA is primarily a problem of categorizing the potential alert into a small number of alert presentation types or modalities. We need to determine when the VOA crosses thresholds (defined by the VOI/VOA specification) indicating, for example, that it is valuable to issue an alert, or that the alert should be issued as high priority. In our framework, the thresholds are customizable by the user and can be mission specific, so they can change automatically as different missions in the plan are executed. The VOI algorithms also determine what information to include in an alert.

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

Different alert presentations are handled by assigning a qualitative *priority* to each alert. We divide alerts by VOA into four equivalence classes for levels of priority. Each priority is presented differently to the user, from using different modalities to simply using different colors or sizes of text or graphics. These priority levels can be used to adjust alerting behavior to the user's cognitive load. For example, during fast-paced operations, only the highest-priority alerts could be presented.

There are several reasons for preferring qualitative reasoning, and we draw on Forbus's work in describing the advantages (Donlon and Forbus, 1999; Forbus, 2002). Qualitative models fit perfectly with making decisions, which are discrete events, and effectively divide continuous properties at their important transitions. Thus, changes in qualitative value generally indicate important changes in the underlying situation. Our qualitative models facilitate communication with humans because they are built on the reasoning of human experts and thus are similar to people's understanding. They are also useful for integrating the results of various qualitative computations in a way humans can understand. Finally, the false precision of quantitative models can be a serious weakness (e.g., they may provide a false sense of security) if the underlying models do not have sufficient accuracy. Common-sense reasoning about continuous quantities is often done qualitatively. The continuous value is of interest only when a different action or decision is required. For example, you can ignore your fuel gauge when driving once you have decided whether or not you must refuel before reaching your destination.

The Executive keeps alert histories for each actor that it alerts. Our VOA calculations take into account the frequency and timing of alerts that have already been given, thus avoiding the problem of cascading alerts. In addition, alerts "expire" in the sense that they can no longer be used to suppress future alerts. The idea is that a human actor may have forgotten information provided too far in the past. More details on our solution to the challenges of monitoring can be found elsewhere (Wilkins et al., 2003).

2.2. Dynamic Scheduler

The DS provides capabilities for assessing the broader impact of events that have caused alerts and for determining appropriate mitigating changes to the current airlift schedule. As indicated earlier, the DS extends the technology and software first implemented in the AMC Allocator (Kramer and Smith, 2002; Smith et al., 2004), a system for day-to-day management of airlift and tanker schedules that is now embedded as an operational module in the AMC Consolidated Air Mobility Planning System (CAMPS) mission-planning system. The AMC Allocator utilizes incremental, constraint-based scheduling techniques that allow selective reoptimization of allocation decisions to accommodate new higher-priority missions while minimizing disruption to previous assignments.

The AMC Allocator primarily addresses the problem of allocating available aircraft and aircrews to pending missions. As resource assignments are made to a given mission, the system also manages the generation and insertion into the mission plan of any necessary auxiliary tasks (e.g., positioning flights or scheduling crew rest periods), to ensure that all relevant constraints are satisfied and the mission is executable. In the simplest case, all missions are planned and scheduled as round-trips. Various missions will be sequenced when necessary to satisfy overall resource capacity constraints, and since there are always more potential missions that could be flown than current aircraft and aircrew levels will allow, some subset of lower-priority missions will invariably be rejected as unsupportable.

In those cases where available aircraft or aircrew capacity does not exist to support a given mission, the system can be invoked in what-if mode to selectively relax constraints and generate various options for accommodating the mission. For example, the system can produce options that involve *delaying* the mission (i.e., relaxing its latest arrival date) until there is available resource capacity at one or more air wings; options that involve *over-allocating* the aircraft or aircrew capacity of a given air wing (in which case, it is assumed that an agreement will be reached to "borrow" an asset that has been held back by the wing for local purposes); options that involve *bumping* a lower-priority mission in the schedule (in which case, an attempt is made to reallocate this mission); or options that involve combinations of many of these basic options.

It is also possible to direct the system to consider *mission merging*, which provides another means for optimizing resource usage. For example, the system might suggest using an aircraft from one mission to support a second mission instead of returning directly back to home station.

Human planners interact with the AMC Allocator through graphical displays, which incorporate mission-oriented, resource-oriented, and map-based views of the current set of commitments. More details on the search techniques underlying this approach to scheduling can be found elsewhere (Smith et al., 2004).

2.2.1. Solving schedule management challenges

Although it is currently utilized in "planning mode" to integrate new pending missions into the global AMC airlift and tanker schedule, the functional capabilities of the AMC Allocator are equally relevant to execution management and this fact has motivated the development of the DS. One difference in an execution management context is the need to accept and respond to so-called "state of the world" updates—new information about the status of missions and the availability of air assets—and several architectural extensions were made to our scheduling technology to meet this requirement. A message-handling module was incorporated to interact with external information sources.

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

In particular, the FMA Executive (via OAA) sends messages reflecting various alerts that have been triggered.

Upon receipt of a message (alert) by the scheduler, a new "schedule update" mechanism is invoked to reconcile the newly received status information with the expectations contained in the current schedule and determine those effects (e.g., conflicts in the schedule, opportunities for further optimization) that require response. An *Agenda Panel* was added to the scheduler's GUI for displaying, managing, and examining the effects of alerts received from the FMA Executive. Graphical tools were also developed for visualizing the impact of an alert on the existing schedule. The overall DS architecture is depicted in Fig. 2.

Application of our scheduling technology in execution management has also required infrastructure changes to the scheduler's core constraint representation and search techniques. On one hand, more detailed resource and constraint models are needed for effective rescheduling at execution time. For example, various MOG constraints at different ports, which dictate limits on how many aircraft can be simultaneously on the ground, must be modeled and accounted for. Likewise, it is necessary to model mission itineraries with greater fidelity, including the tracking of such ground activities as takeoffs, blockins, preflights, and postflights.

These more detailed resource and constraint models were accommodated in part by introducing an underlying Simple Temporal Network representation (Dechter et al., 1991) of mission itineraries (replacing the more traditional "fixed-times" schedule representation of the original AMC Allocator). By moving to this flexible-times representation of schedules, certain prerequisite types of repairs to execution problems become easily supported. For example, dynamic extension of support activities such as crew rests can easily be used to accommodate resource conflicts.

The DS is designed to support mixed-initiative scheduling, allowing the end user a range of interaction options. These options range from primarily manual with constraint checking, to user selection of system-recommended options for schedule deconfliction, to fully automated rescheduling actions based on predefined user preferences. The DS incorporates all previously developed options for relaxing constraints in circumstances of constraint conflict (e.g., mission delay, overallocation of a wing) and also introduces some new options more relevant at mission execution time. In particular, to resolve problems that involve in-process missions, the DS computes options that include ground-activity delay and diversion to an alternative landing location.

Section 3 provides a detailed example of a mixed-initiative schedule repair.

2.3. Dynamic Mediator

The DM provides an option for the human flight manager to make an effective decision by gathering information from other actors quickly. When the flight manager must alter the schedule in response to an unexpected event, time is an important factor because a delayed decision may require the schedule to be altered even more. For example, when faced with a reduction in MOG capacity, the flight manager needs to make a decision that allocates the remaining capacity to the missions that most require it.

The DM module makes two main assumptions: (1) no single entity possesses all the information relevant to the decision and (2) the time allowed for making the decision is limited or a delayed decision is costly.

The originating planners have information relevant to making alterations to the mission schedule. For example, when faced with a reduction in capacity, the flight manager wants to make a decision that allocates the remaining capacity to the missions that most require it. Relevant information held by the human planners includes, for example, the cargo contents and the purpose of the mission. However, this information is not normally entered into the FMA (nor the existing AMC software tools that support execution) because it is not needed to generate schedules.

Extracting information relevant to decision making is costly because planners must be contacted to extract information. The DM automates parts of the process of

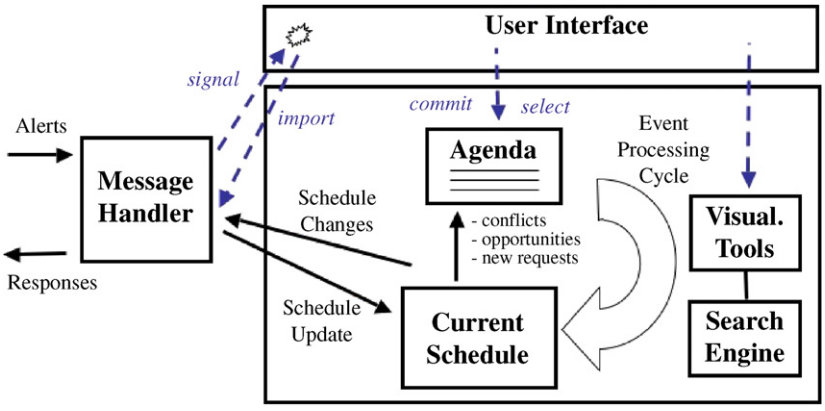

Fig. 2. Internal architecture of the Dynamic Scheduler.

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

incrementally extracting only that information that is relevant to the flight manager's decision. The DM lowers the cost of collecting information and computing the correct decision. The grouping and ordering of information-extracting questions is organized in such a way as to minimize the cost of extraction while ensuring that the decision is made correctly.

Fig. 3 provides a decision tree describing an instance of a problem that may be handled by the DM. This problem instance involves two originating planners labeled P_1 and P_2. P_1's mission is labeled m_1 and P_2's mission is labeled m_2. The flight manager must prioritize the two missions. Each mission has attributes relevant to the flight manager's decision that are known to the planners but not the flight manager. In this example, we assume that the only relevant attribute is the type of cargo. P_1's mission has possible cargo types a_1, a_2, and a_3. P_2's mission has possible cargo types b_1, b_2, and b_3. The leaf of the tree in the diagram indicates the mission that the flight manager wishes to choose, given the cargo type of each mission labeled on the edges from the leaf to the root of the tree. For example, with cargo types (a_2, b_3), the flight manager chooses mission m_2.

The diagram provides a sequence for information extraction used by the DM based on minimizing the expected number of queries. First, planner P_1 is queried for its cargo value, and then planner P_2 is queried for its cargo value. Cost savings in terms of minimizing the number of questions are achieved because of the structure of the problem. For instance, if planner P_1's cargo is either a_1 or a_3, no query of planner P_2 is required. On the other hand, if P_2 was the first planner queried, P_1 would need to be queried regardless of P_2's cargo type.

Prior to the FMA, communication between flight managers and planners was attempted in only the most important situations because interpersonal communication was too costly. As a result, the flight manager often makes an educated guess as to the attributes of the relevant missions and a decision based on that guess may be inappropriate. The DM makes communication practical by

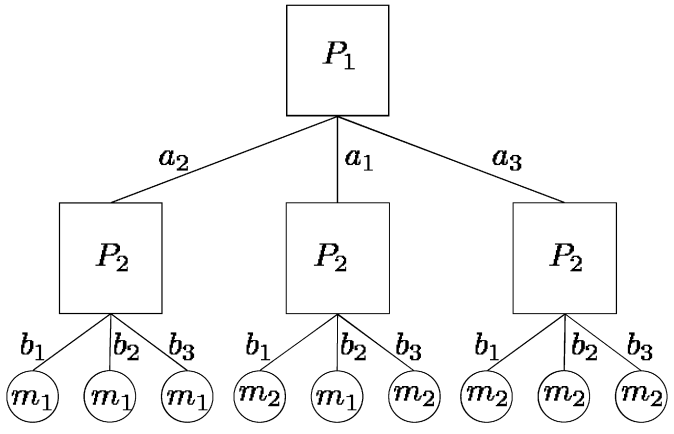

Fig. 3. Dynamic Mediator: example problem input and information extraction sequence.

(1) managing the communication between the flight manager and the planners to focus on relevant information and (2) storing, organizing, and analyzing the information for the purpose of making a decision. The DM module enables the flight manager to make better decisions during execution, while not precluding the use of personal contact for the most important decisions.

The DM module automates collection of relevant information from planners using queries and replies, implements a search for those queries and replies that minimize the expected communication costs, and enables correct decision making with limited information.

3. The FMA in operation

The FMA was demonstrated and evaluated on several vignettes, using data feeds similar to actual AMC data feeds. These vignettes are described in Section 4. In this section, we give an indication of the operation of the FMA by selecting one particular vignette and describing the execution flow in some detail.

In this vignette, multiple events (bad weather and an Instrument Landing System (ILS) failure), when considered together, cause a problem, although neither event alone would be problematic. The FMA detects the problem and suggests responses. An overview of the events and responses in this vignette follows.

(1) The Executive receives a report that the ILS for port P will be offline for a time window $[t1, t2]$ for repairs.
(2) The Executive receives a weather exception at P that overlaps with $[t1, t2]$.
(3) The Executive infers that the airport will be closed for some period due to simultaneous bad weather and no ILS capability. Either event by itself is no problem but together they cause a problem, and the domain-specific knowledge in the VOI calculation recognizes this interaction.
(4) The Executive communicates port closure information to the scheduler.
(5) The Executive queries the Scheduler for affected missions and alerts the Execution user and affected planning offices, customizing the alert to each actor (assuming the VOI and VOA are sufficiently high).
(6) The Scheduler automatically computes immediate impact and suggests rescheduling actions.
(7) The Execution user, possibly collaborating with planning offices using the DM, selects a schedule fix, after possibly modifying it during interactions with the Scheduler.
(8) The Execution user and appropriate planning offices are notified of all relevant changes to missions.

After the FMA detects the problem, it issues alerts (in Event 5 above) with flash (highest) priority to the human flight manager and to any planning actors that are affected. Fig. 4 shows a notional alert interface we developed for

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

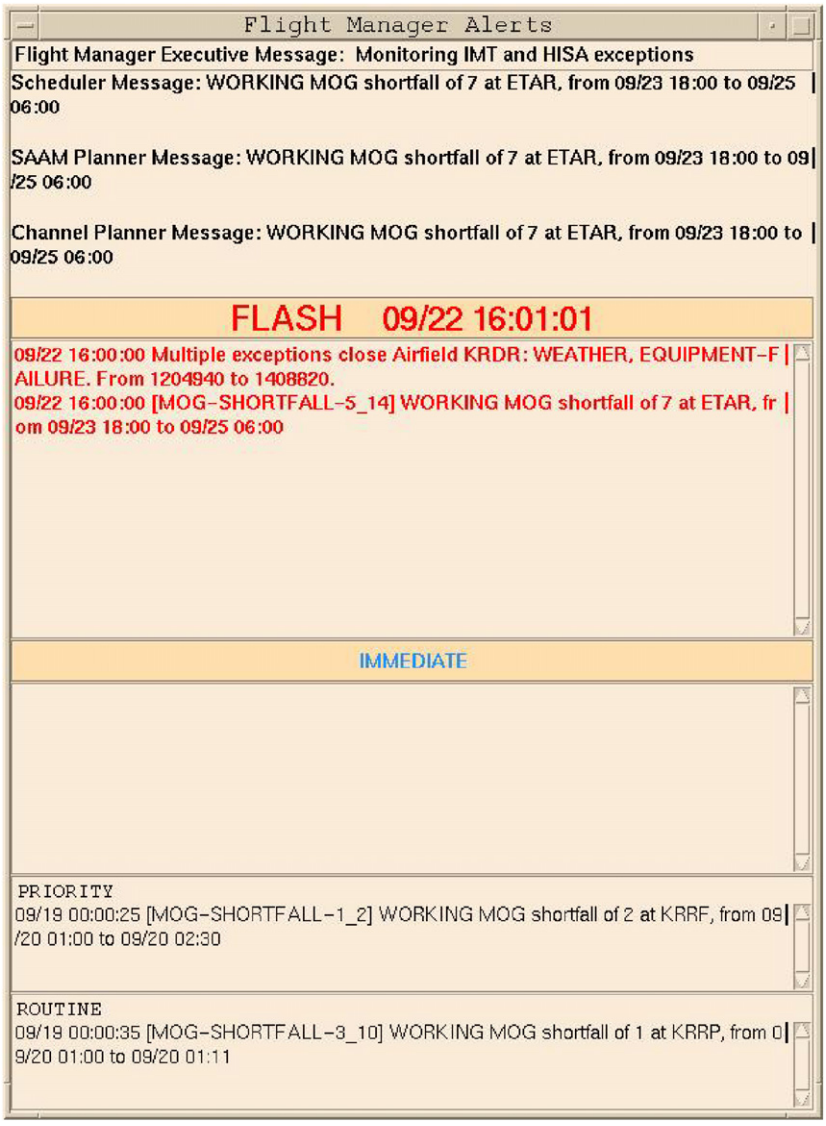

Fig. 4. Alerts presented to the flight manager by the Executive.

demonstration and evaluation purposes. The top window shows messages that have been sent to other actors. The red FLASH window shows the alert we are discussing first, and another unrelated alert that was detected at the same time, although it concerns events that will happen the following day. Finally, two lower-priority alerts can be seen in the bottom windows.

At the same time these alerts are give, the Executive also sends a message with the details of the detected problem to the DS. We now look in detail at Event 6, working through a small case study of how the DS identifies missions that are in conflict, and how the user employs it to find solutions to these conflicts. Fig. 5 depicts the DS "Agenda Panel" after an alert has been received from the FMA Executive. The current time is 09:00 on October 4, 2004, and the base closure at KSUU—Travis Air Force Base in California—is predicted to last from 12:00 to 22:00. The missions affected by this base closure are either scheduled to takeoff or to land at Travis during that 10-h period. While it is possible that this set could by empty, it is highly unlikely.

When the user asks to examine this base-closure event, the system computes that seven missions are scheduled to takeoff or land at Travis during the event interval, and presents them to the user as shown in Fig. 6. The user may examine each of these missions in turn and work with the DS to find alternatives for deconflicting them. Alternatively, the user could direct the system to find the best overall deconfliction strategy automatically, based on various preferences and rules.

For this conflict, all seven missions must be modified, but in other conflict situations deconflicting some subset of involved missions would be sufficient. For instance, suppose the delayed arrival of a mission has caused an air base to have only 10 aircraft when 11 missions are scheduled. While 11 missions are involved in this conflict, only one need be rescheduled to solve the conflict. In such cases, the DS will suggest the best subset of missions to

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

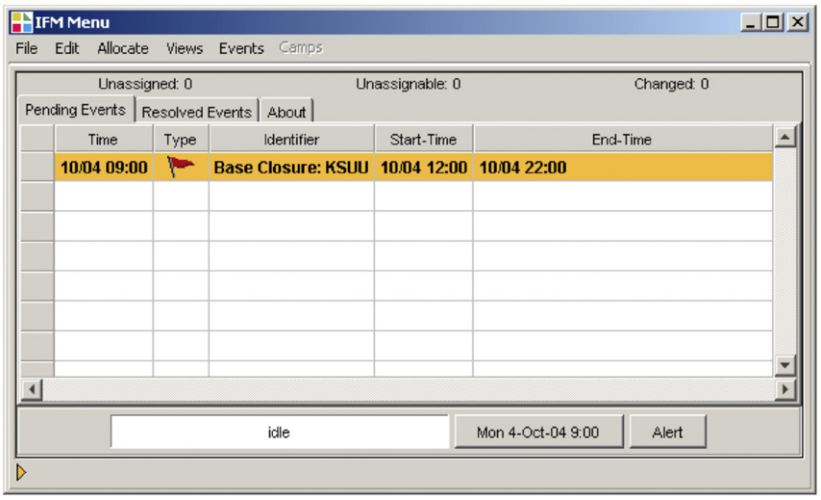

Fig. 5. The Dynamic Scheduler agenda panel.

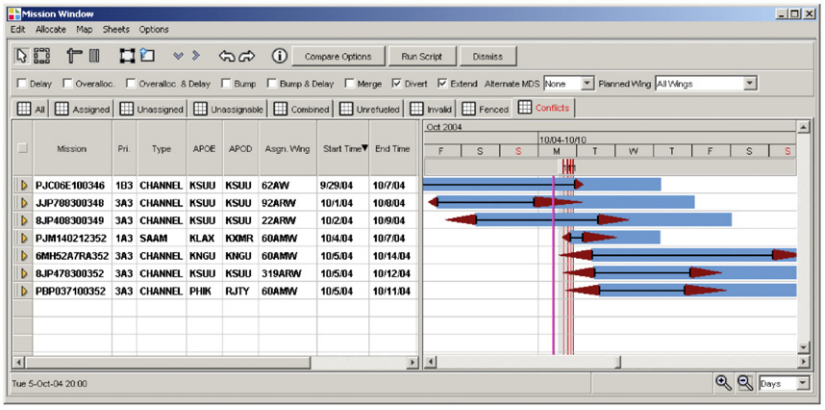

Fig. 6. Conflicted missions.

reschedule based on factors such as mission priority and other user preferences.

In Fig. 7, we have drilled down on the third of the conflicted missions to display all the activities in its itinerary. The current time, 09:00, is depicted by the vertical line to the left in the Gantt display, indicating that much of the mission has already been executed. The shaded region to the right of the current time indicator depicts the Base Closure interval of 12:00–22:00. Note that the mission is scheduled to transit from RJTY (Yokota Airbase in Japan) to KSUU (Travis Air Force Base) during this time interval. The mission does not come into conflict until the end of this transit interval, when a touchdown activity is scheduled. This conflict is indicated by the second vertical line immediately before the "postflight and offload at KSUU" activity.

Because the current time of 09:00 corresponds with the beginning of the Yokota-to-Travis leg of the mission, the options for deconflicting it are very limited. Given the latency in communications, it is likely that the mission has already departed. Accordingly, we use the system to query possibilities for diverting the mission en route to an intermediate location. In computing diversion options, the scheduling engine seeks to find those diversions that minimize additional flying distance while avoiding the Base Closure interval.

As can be seen in Fig. 8, the DS has presented the user with four options for diversion—PADK, PADU, PASY, and PATU—all locations in the Aleutian Islands of Alaska. From the map display it is clear that they are all close to the great circle route that the mission would take from Yokota to Travis.[1] Note that all of the options generated (the system presents just the top four) will cause a delay of 16 h in the mission, much more than it would need to avoid the base closure at Travis, and actually much more than the extra time necessary to fly to the intermediate location, refuel, and take off. Most of the extra 16 h is due to an extra "crew rest" activity that the system has inserted to comply with maximum crew-duty day rules.

[1]A great circle route represents the shortest distance from one location to another on a globe, and while missions do not generally follow exact great circle routes, these routes represent an approximation to the actual route.

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

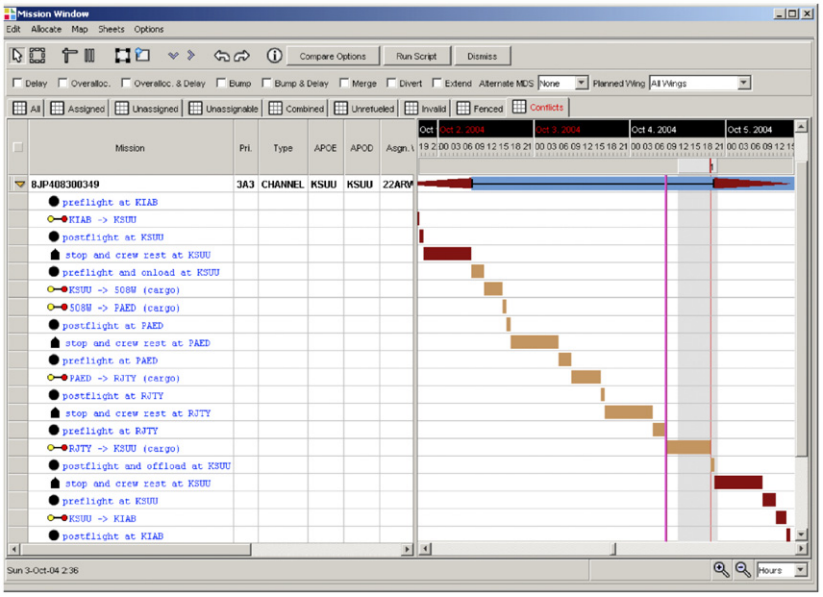

Fig. 7. Detailed itinerary for one mission.

In Fig. 9, we see the modified itinerary after the user has selected the PADU—the airbase at Unalaska—diversion. Thus, there is a valid solution to deconflict this particular mission. Similar strategies can be used to deconflict the remaining six missions, depending on whether or not they are executing. One other deconfliction option is to "Extend" a mission activity, for instance a scheduled crew rest, in advance of the conflict interval. As with the diversion options, the user can direct the DS to compute Extend options that minimize mission delay, yet avoid a conflicted time interval.

Missions that are not yet in execution can be deconflicted by employing a number of rescheduling options such as Delay, Overallocate, and Bump. These options are described elsewhere (Kramer and Smith, 2002; Smith et al., 2004).

4. Evaluation

We defined a demonstration scenario consisting of several storyboard-level vignettes that illustrate the capabilities of the FMA. These vignettes were developed in conjunction with subject-matter experts and guided the design of the FMA. Our evaluation consisted primarily of demonstrating FMA capabilities for these vignettes and evaluating performance.

The FMA was demonstrated on the vignettes using scripted data feeds that were generated to be as similar to actual data feeds as possible. For instance, one such script uses all 1100 MOG exceptions from the output of HISA. Based on review by subject-matter experts, all the demonstrated vignettes show useful capabilities beyond what is currently provided by existing AMC flight-management software tools.

4.1. Vignettes

A brief summary of each vignette follows. The third vignette was described in detail in Section 3.

- A MOG conflict is detected by the Executive and resolved by the execution office and planners with assistance of the DS.
- A single event causes multiple, cascading problems. An airplane breaks on the runway of Airport 1, causing both a problem with lack of aircraft resources and a problem with stalled cargo. The Executive detects the problems, and DS-aided responses must handle multiple problems.
- Multiple events (bad weather and an ILS failure), when considered together, cause a problem, although neither event alone would be problematic. The FMA detects the problem and suggests responses.
- The FMA monitors system behavior and gives alerts or responds to the situation. For example, FMA might alert when AMC tools that report MOG exceptions and execution-time exceptions are not present or have lost input feeds, or when FMA actors are not present.
- The FMA performs automated responses to a minor problem, controlled by user-established and selected policies.

4.2. Performance

We measured the time required for generating alerts and schedule repairs, and in all cases the FMA response time was more than adequate for AMC requirements. The alert detection is nearly instantaneous. Schedule repairs are discussed later in this section. We had subject-matter experts subjectively evaluate the performance of the FMA.

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

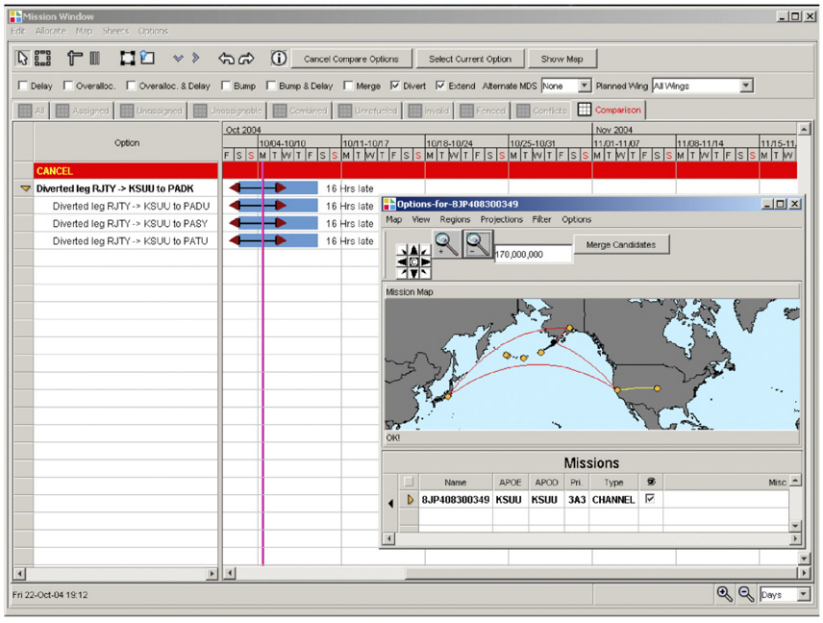

Fig. 8. Options for deconfliction.

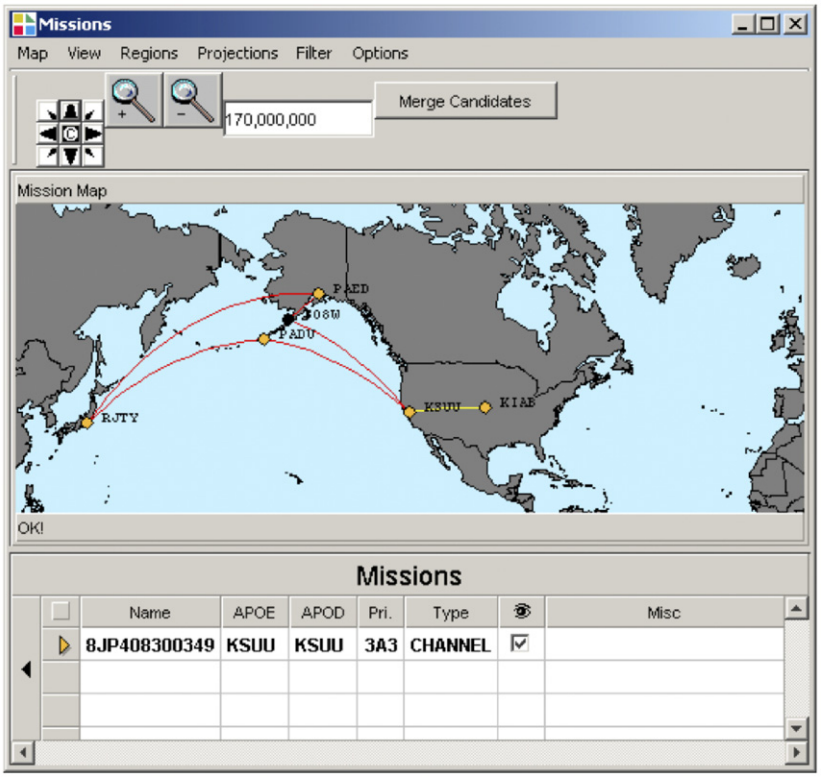

Fig. 9. The revised itinerary.

They determined that the alerts generated and schedule repairs completed using the FMA were correct and valuable in each of the vignettes.

The Executive (1) monitors all exceptions from multiple tools, (2) estimates the value of each possible alert, and (3) issues high-value alerts that focus user attention on key problems. Using actor-specific VOA, it effectively filtered and prioritized the alerts generated by existing AMC tools. For example, we ran the Executive and SAAM actors on 1085 actual MOG alerts, which were generated by an AMC

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

software tool. The Executive assessed the significance of each, and filtered all but 242 of the 1085 alerts, of which only one was highest priority, and only eight required immediate attention (the second-highest priority level). The Executive sent the SAAM actor 145 alerts, all of which were lower priority.

The overall plan used in our vignettes had approximately 1000 missions comprising 3000–4000 sorties (individual flights) over a 2- to 3-week horizon. The DS is designed to support interactive resolution of any problems that are introduced into the current schedule as a result of importing the information contained in a given alert. The repair actions, of course, need to repair only a small section of the entire plan, from a few missions to several dozen missions.

For most repairs, the DS took only a few seconds to generate options for the human to explore. Our most complex repair involved resolving a full base closure with 20–30 missions in conflict, which required adjustments to at least seven missions (of around 1000 in the global schedule). Alternatively, the scheduler can be scripted to fully automate this conflict-resolution process and in this mode the DS is also quite efficient, producing its best solution in a time frame of several seconds (less than 30 s).

Most of our evaluation vignettes involved machine/human collaboration. The user drives exploring and selecting among options for responding to the schedule conflicts caused by an event. In such cases, the overall time was dominated by the human's response. That is, the DS responded in real time and the time used by the DS was not a limiting factor.

4.3. Application status

The FMA is designed to coexist with and complement the existing flight-management software tools currently deployed at AMC. Some existing tools at AMC detect deviations and problems, but they are based on simple rules. They detect too many false alarms that overwhelm the user with alerts so that the user cannot focus on the most important deviations. The FMA improves upon these tools by its VOA computation, which will filter out low-value alerts, and show high-value alerts to those users for whom they have high value. Furthermore, the FMA detects problems that are not detected by existing tools (e.g., the closure vignette described above).

The FMA has been demonstrated as a proof of concept, but is not used in actual operations, although the DS is already in use at AMC as part of CAMPS. The Executive generally takes inputs in forms that are available in existing AMC tools and databases. The DM work is more preliminary, but the FMA can be transitioned without the DM. The primary tasks that would be required to transition this technology for daily use are as follows:

- The Executive must integrate and interface with any data sources to be monitored (it may be desirable to monitor additional data sources).
- The FMA system operates in real time, but must be made more robust with respect to tracking and reasoning about current time.
- Design and implementation of an interactive alert/collaboration GUI or integration with existing GUIs must be accomplished.
- Policies must be encoded to implement AMC procedures regarding schedule repairs.

5. Comparison to other work

5.1. Monitoring

FMA is designed to coexist with and complement the existing AMC flight-management software tools. Some existing AMC tools use simple rules to detect deviations and problems. They detect too many false alarms that overwhelm the user with alerts and therefore the user cannot focus on the most important deviations.

Plan generation has received a lot of attention in the AI community recently, but rarely are the plans used to control and monitor execution. Even more rarely are plans monitored that involve the activity of hundreds of agents acting simultaneously. Previous work on execution monitoring has focused on models where the executor performs the planned actions (e.g., a robot controller) and usually has direct access to internal state information. In the AMC domain, most actions are performed by external agents, usually humans, and the monitor has no access to the state of its executing agents. Such indirect execution requires different monitoring techniques, as the executor must use incoming messages to determine the status of agents and activities and whether actions have been initiated or completed.

NASA's Remote Agent on Deep Space One (Jonsson et al., 2000; Muscettola et al., 1998) does autonomous execution monitoring on a spacecraft. Our domain has many of the same requirements as NASA's. However, NASA's remote agent is fully automated, which places a heavier burden on the module that generates plans and responses, but alleviates the burden of having to address human interaction issues such as those we consider with VOA. In NASA's domain, the "agents" are mechanical devices onboard the spacecraft, and their behaviors have been formally modeled. Our agents include humans, whose behaviors are not easily modeled, so the FMA estimates the value of alerts and interacts with a human decision maker, who ultimately is responsible for the control decisions.

The general monitoring approach used by the Executive is also compared to a variety of other monitoring approaches in Wilkins et al. (2003).

5.2. Schedule repair

The DS uses an incremental, mixed-initiative approach to managing the schedule during execution. Information

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

received about the status of executing missions and supporting resources is used first to detect and characterize conflicts (and opportunities) in the current schedule, and then option generation/schedule revision capabilities are iteratively employed to explore the downstream impact of prospective corrective actions. In the AMC context, Wampler et al. (2005) have also explored the use of graphical visualizations for conveying the impact of execution events on future operations, however without an underlying schedule management infrastructure.

In the arena of space-mission planning, the MAPGEN system (Bresina et al., 2005) provides a similar constraint-based infrastructure for manipulating schedules, along with graphical aids for conflict detection and analysis. However, whereas the DS provides a range of incremental schedule revision capabilities, the MAPGEN engine operates principally with regenerative (and hence more disruptive) scheduling techniques and is hence better suited to advance planning applications.

The development of techniques for incrementally managing schedules in response to execution status updates has received attention in the AI community (Smith, 1994; Zweben et al., 1994; Bierwirth and Mattfeld, 1999; Chien et al., 2000; El Sakkout and Wallace, 2000; Kramer and Smith, 2004). The approach taken in the DS descends from Smith (1994) and combines the use of strong domain heuristics with local repair-based search techniques (Kramer and Smith, 2004) to strike a balance between optimization and solution stability. Other work (e.g., Zweben et al., 1994; Bierwirth and Mattfeld, 1999; Chien et al., 2000), places greater reliance on broader search-based processes, which also provides a basis for efficient, "anytime" scheduler response but with less emphasis on minimizing change.

From an application perspective, the CASPER system at NASA's Jet Propulsion Laboratory (Chien et al., 2000) is designed to solve much the same kind of problem as the DS. It uses an iterative-repair scheduling approach to maintain space-mission schedules in response to updates in mission status, and has been successfully applied to manage sensing activities in a continuous, closed-loop manner in the recent Earth Observatory 1 space mission. Overall, however, the domains addressed by CASPAR and DS tend to be rather different in scale and character. The AMC problem requires management of hundreds of assets over a short-term horizon; the EO-1 problem alternatively involves much smaller numbers of resources but requires attendance to potentially more idiosyncratic action sequences.

5.3. Dynamic mediation

The aim of the DM is to minimize human effort in group decision making. Prior work has investigated metrics for measuring effort in decision making among automated agents. Several recent advances in multiagent algorithm design are motivated primarily by the need to minimize communication bandwidth requirements (Davin and Modi, 2005; Mailler, 2005; Rauenbusch, 2004). Sunderam and Parkes (2003) focused on measuring effort in communication of an agent's private information (Sunderam and Parkes, 2003). The DM was motivated by these techniques developed for agents and demonstrates their application to the interactions between flight managers and planners.

6. Conclusion

The FMA facilitated the use of distributed expertise to monitor large amounts of execution data, detect problems, and quickly modify the plan. This general conclusion is based on three more specific conclusions.

- The approach of using domain knowledge to model the VOI and value of alerts allowed the FMA to give only high-value, user-appropriate alerts.
- The actor architecture provided flexibility and allowed the FMA to partially automate and regulate information flow and communication according to the overall global situation, and to each human's preferences and current workload. For example, a planning actor gets a high-priority alert only when additional information is needed to replan a mission owned by the actor.
- The scheduling approach provided the plan representation used by all actors to organize available information so that decision making could be supported by using software agents in tandem with, and specific to, their human counterparts. The Scheduler, either automatically or in a mixed-initiative interaction with humans, used this representation effectively and rapidly modifying the plan during execution, when the Executive detected problems.

There are often no obvious boundaries to the types of support an execution aid might provide in a real-world domain. We believe the FMA demonstrates an effective balance between the capabilities provided and resources used for domains similar to the AMC domain. We conclude by describing the benefits of our approach.

By distilling thousands of exceptions detected by AMC software tools into a handful of high-priority alerts, the FMA Executive greatly reduces the amount of information humans must monitor, allowing the humans to concentrate on more important tasks than monitoring large amounts of incoming information.

Because humans are immediately alerted to problems and have automated support in constructing repair options, they make both faster and better responses to unexpected events. Better responses refer to some combination of accomplishing more missions, using resources more efficiently, violating fewer constraints, and causing fewer downstream problems to future missions. Faster responses are enabled by both the immediate recognition of the problems and the fast, automated generation of repairs.

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

The automated support means that large, complex schedules can be accurately monitored and that no relevant information is ignored or missed. Distributed actors get actor-specific alerts that keep them apprised of the status of their missions and allow them to provide feedback during execution.

The Dynamic Scheduler (DS) provides a range of capabilities for interacting effectively with the human decision maker. Upon receipt of an alert from the Executive, the status information contained in the alert is superimposed over the executing schedule, and a list of resulting issues (e.g., schedule conflicts) is posted on an agenda panel. As the user selects a given conflict, the system invokes graphical displays that indicate the impact. The DS can be directed by the user to generate alternative repairs for the selected conflict. Importantly, policies control system responses. Thus, if policy permits, the DS can be invoked automatically by the Executive. Decisions are communicated back to the Executive for implementation.

The coupling of intelligent execution monitoring to dynamic schedule repair has enabled more efficient and rational global flight management.

Acknowledgments

This research was supported by the Air Force Research Laboratory, Rome under Contract F30602-02-C-0152. We thank Steve Hofmann of MITRE for his domain expertise and advice.

References

Bierwirth, C., Mattfeld, D., 1999. Production scheduling and rescheduling with genetic algorithms. Evolutionary Computation, 7 (1).

Bresina, J., Jonsson, A., Morris, P., Rajan, K., 2005. Activity planning for the Mars exploration rovers. In: Proceedings of the International Conference on Automated Planning and Scheduling, Monterey, CA.

Cheyer, A., Martin, D., 2001. The Open Agent Architecture. Autonomous Agents and Multi-Agent Systems 4 (1–2), 143–148.

Chien, S., Knoight, R., Stechert, A., Sherwood, R., Rabideau, G., 2000. Using iterative repair to improve the responsiveness of planning and scheduling. In: Proceedings of the International Conference on Automated Planning and Scheduling. Breckenridge, CO.

Davin, J., Modi, P.J., 2005. Impact of problem centralization in distributed constraint optimization algorithms. In: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems.

Dechter, R., Meiri, I., Pearl, J., 1991. Temporal constraint networks. Artificial Intelligence 49, 61–95.

Donlon, J., Forbus, K., 1999. Using a geographic information system for qualitative spatial reasoning about trafficability. In: Proceedings of the Qualitative Reasoning Workshop. Loch Awe, Scotland.

El Sakkout, H., Wallace, M., 2000. Probe backtrack search for minimal perturbation in dynamic scheduling. Constraints 5 (4).

Forbus, K.D., 2002. Towards qualitative modeling of the battlespace. Technical Report unpublished manuscript, Northwestern University, Evanston, IL.

Jonsson, A., Morris, P., Muscettola, N., Rajan, K., 2000. Planning in interplanetary space: theory and practice. In: Proceedings of the International Conference on Automated Planning and Scheduling, Breckenridge, CO. AAAI Press, Menlo Park, CA, pp. 177–186.

Kramer, L., Smith, S., 2002. Optimizing for change: mixed-initiative resource management with the AMC barrel allocator. In: Proceedings of the Third International Workshop on Planning and Scheduling for Space, Houston.

Kramer, L., Smith, S., 2004. Task swapping: a broader analysis. In: Proceedings of the International Conference on Automated Planning and Scheduling. Whistler, CA.

Mailler, R., 2005. Comparing two approaches to dynamic, distributed constraint satisfaction. In: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems.

Mulvehill, A., Whitaker, R., 2002. Human interaction with software agents (HISA). Technical Report, Air Force Research Laboratory.

Muscettola, N., Nayak, P.P., Pell, B., Williams, B.C., 1998. Remote agent: to boldly go where no AI system has gone before. Artificial Intelligence 103 (1–2), 5–47.

Ortiz, C.L., Rauenbusch, T.W., Hsu, E., 2003. Dynamic resource-bounded negotiation in non-additive domains. In: Lesser, V., Ortiz, Jr., C.L., Tambe, M. (Eds.), Distributed Sensor Networks: A Multiagent Perspective. Kluwer Academic Publishers, Dordrecht.

Rauenbusch, T.W., 2004. Measuring information transmission for team decision making. Ph.D. Thesis, Harvard University.

Smith, S., 1994. Opis: a methodology and architecture for reactive scheduling. In: Zweben, M., Fox, M. (Eds.), Intelligent Scheduling. Morgan Kaufmann, Los Altos, CA.

Smith, S.F., Becker, M.B., Kramer, L.A., 2004. Continuous management of airlift and tanker resources: a constraint-based approach. Mathematical and Computer Modeling—Special Issue on Defense Transportation: Algorithms, Models and Applications for the 21st Century 39 (6–8), 581–598.

Sunderam, A.V., Parkes, D.C., 2003. Preference elicitation in proxied multiattribute auctions. In: Proceedings of the Fourth ACM Conference on Electronic Commerce.

Tsien, C., 1997. Reducing false alarms in the intensive care unit: a systematic comparison of four algorithms. In: Proceedings of the American Medical Informatics Association Annual Fall Symposium.

Tsien, C., Fackler, J., 1997. Poor prognosis for existing monitors in the intensive care unit. Criitical Care Medicine 25 (4), 614–619.

Wampler, J., Whitaker, R., Roth, E., Scott, R., Stilson, M., Thomas-Meyers, G., 2005. Cognitive work aids for C2 planning: actionable information to support operational decision making. In: Proceedings of the Tenth International Command and Control Research and Technology Symposium.

Weinberger, E., 2002. A theory of pragmatic information and its application to the quasispecies model of biological evolution. Biosystems 66 (3), 105–119.

Wilkins, D.E., Lee, T.J., Berry, P., 2003. Interactive execution monitoring of agent teams. Journal of Artificial Intelligence Research 18, 217–261.

Zweben, M., Daun, B., Davis, E., Deale, M., 1994. Scheduling and rescheduling with iterative repair. In: Zweben, M., Fox, M. (Eds.), Intelligent Scheduling. Morgan Kaufmann Publishers, Los Altos, CA.

Dr. David E. Wilkins is a Senior Computer Scientist in the Artificial Intelligence Center at SRI International. A fellow of the American Association of Artificial Intelligence (AAAI), he received his Ph.D. in computer science from Stanford University in 1979. His current research focuses on continuous planning and execution monitoring, mixed-initiative and multiagent planning, and policy reasoning for cognitive radios. He is interested in the application of these technologies to real-world problems. His 1988 book, Practical Planning, helped define the AI planning field.

Dr. Stephen F. Smith is a Research Professor in the Robotics Institute at Carnegie Mellon University (CMU) where he is a Director of the Intelligent Coordination and Logistics Laboratory. He received his Ph.D. in Computer Science from the University of Pittsburgh in 1980. His research focuses broadly on the development and use of constraint-based search and optimization techniques for planning and scheduling. He has

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001

directed the development of innovative technologies for applications ranging from military airlift planning to space-based observatory scheduling to manufacturing management. His current research interests include distributed problem-solving, execution-driven planning and configurable systems.

Laurence Kramer is a Project Scientist in the Robotics Institute at Carnegie Mellon University (CMU), where he is a Associate Director of the Intelligent Coordination and Logistics Laboratory (ICLL). He received an MS in Computer Science from the University of Delaware in 1986, after which he focused on software applications of AI, including the long-range planning system of the Hubble Space Telescope. His research interests are in constraint-based search, and AI planning and scheduling systems.

Thomas J. Lee is a Senior Research Engineer in the Artificial Intelligence Center at SRI International. He received his M.S. in computer science from the University of Wisconsin in 1982. His current research includes real-world planning and execution systems, particularly improving their reactivity and their adaptibility through learning.

Dr. Timothy W. Rauenbusch is a Computer Scientist in the Artificial Intelligence Center at SRI International. He received his Ph.D. in Computer Science from Harvard University in 2004. His research focuses on the development and analysis of algorithms for decision making in teams of computer agents and the development of computer systems that support negotiation among people. As an undergraduate at the University of Pennsylvania, Tim wrote the Quick Start Manual for Eniac, the world's first general-purpose digital computer.

Please cite this article as: Wilkins, D.E., et al., Airlift mission monitoring and dynamic rescheduling, Engineering Applications of Artificial Intelligence (2007), doi:10.1016/j.engappai.2007.04.001