

Optimizing for Change: Mixed-Initiative Resource Allocation with the AMC Barrel Allocator

Laurence A. Kramer and Stephen F. Smith

The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213

{lkramer, sfs}@cs.cmu.edu

Abstract

Most real world planning and scheduling domains are only partially amenable to optimized solutions due to size of the solution space and due to the impossibility of fully optimizing a problem that changes over time. In addition, even an “optimal” solution to a problem is of little use if it is not comprehensible to end users of the system. In this paper we describe the AMC Barrel Allocator, a resource allocation tool for day-to-day airlift and air refueling management for the USAF Air Mobility Command (AMC). We discuss the design elements that promote user management of the solution space in terms of volume, complexity, and change. This work has relevance even for domains where fully automated operational usage is envisaged, but where system validation and user acceptance are issues.

Introduction

Optimization of resource allocation in the domain of airlift and air refueling planning and scheduling has historically been somewhat lacking. For the past two years the AMC Barrel Allocator has been an operational module in AMC’s Consolidated Air Mobility Planning System (CAMPS), where it aids in optimizing resource allocation in a flexible and iterative fashion. We describe how the design of the AMC Barrel Allocator is “optimized for change” along several dimensions: helping users to change from an old, mostly manual mode of work to a new, mostly automated one, and incorporating optimization into an environment that is subject to change on a daily basis.

Work in the area of mixed-initiative planning and scheduling is generally defined in terms of a division of labor into those areas where the computer excels – for instance extensive search through alternative possibilities to find a desired solution – and those where human input is valuable – synthesizing and decision making at higher levels of abstraction. [Burststein and McDermott, 1994]. Recent research [Cox and Veloso, 1997. Scott, Lesh, and Klau, 2002] indicates that mixed-initiative systems should

be designed to be flexible in the boundary between where the user and where the computer has the initiative, based on the differing user abilities and interests. This viewpoint is amplified in [Rich, Sidner, and Lesh, 2001] who argue for casting human-computer interaction as a collaborative dialogue process that behooves all intelligent interfaces to support certain basic functionality.

We argue that a mixed-initiative approach to scheduling and resource allocation is important even in areas where the computer could do an excellent job if left untended, since it allows users to gain trust in the system. Users may well be impressed at the speed with which the program can perform routine tasks, but will remain reticent to use the system unless they can question the system’s decisions, and perform what-if analyses to explore multiple solutions to a problem. [Smith, Lassilia, and Becker, 1996]

In the first section of the paper we present a brief description of the domain, the architecture of the Barrel Allocator, and some changes to its original design. In section two we introduce the Barrel Allocator interface, and in succeeding sections present a case study that illustrates a continuum of system tools that allows for resource allocation ranging from manual to semi-automated to fully automated. Finally, we summarize the benefits of mixed-initiative interfaces and suggest some future work.

The AMC Barrel Allocator

The AMC Barrel Allocator [Becker and Smith, 2000] is a tool that helps Air Force and civilian planners manage resource allocation to thousands of airlift missions and air refueling events over a several week time horizon. Within the Tanker/Airlift Control Center (TACC) at AMC, all planned missions flow into the “Barrel Master’s” office for tasking. The Barrel Allocator is designed to support this tasking process, determining which air wing will be assigned to fly any given mission. An air wing assignment specifies an air crew and an aircraft of a particular type at a particular base. The allocation objective is to minimize

non-productive flying time and support as many high priority missions as possible, while respecting such constraints as mission start and end times, aircraft flight range, and wing capacities.

This allocation problem is typically oversubscribed, and the system is designed to incorporate new missions into a schedule on a continuing basis. Any time that an allocation decision is changed, changes (in the form of revised tasking) must be communicated to affected wings, and hence there is value in minimizing changes to the schedule over time to the extent possible. Through the use of incremental, constraint-based search methods, the system is designed to provide a basis for managing solution change and allows selective re-optimization of allocation decisions as circumstances (e.g., new missions) warrant. The end user is an active partner in this process, utilizing various tools for exploring scheduling scenarios.

A version of the Barrel Allocator was first incorporated in CAMPS in June 2000. Since this initial release, with continuing input from the end users and other domain experts on how to better integrate the Barrel Allocator's functionality with their day-to-day business processes, we've been redesigning some important modules in the Allocator. One particular focus has been better support for allocation of tanker assets (air wings that support air refueling requests). In [Becker and Smith, 2000] the problem of air refueling allocation is described as mainly

one of best-fit "linkage" between already planned airlift missions and pre-planned air refueling missions (say for training purposes). While it is sometimes the case that such opportunistic linkage is employed to match up airlift and refueling missions, in practice the allocation process usually takes the mode of generating new refueling (tanker) missions to meet a fuel demand of receiver missions at a given time and place.

Better understanding of this allocation process has resulted in a reengineering of the air refueling module of the Barrel Allocator, with the redesigned module to go into operational use in June 2002. The redesigned allocation algorithm is similar to that employed for airlift missions with the additional objective of minimizing fuel usage. This is particularly important, as tankers whose mission it is to deliver fuel consume the very product they deliver as they fly to and from the appointed rendezvous point.

We will not go further into system architecture or algorithms in this paper as they are well covered in [Becker and Smith, 2000], but will instead concentrate on the concrete user interface mechanisms employed in the Barrel Allocator which allow end users to manage the volume, complexity, and change inherent in the domain.

The Barrel Allocator User Interface

User interaction with the Barrel Allocator is organized around a spreadsheet-like metaphor – both in terms of a

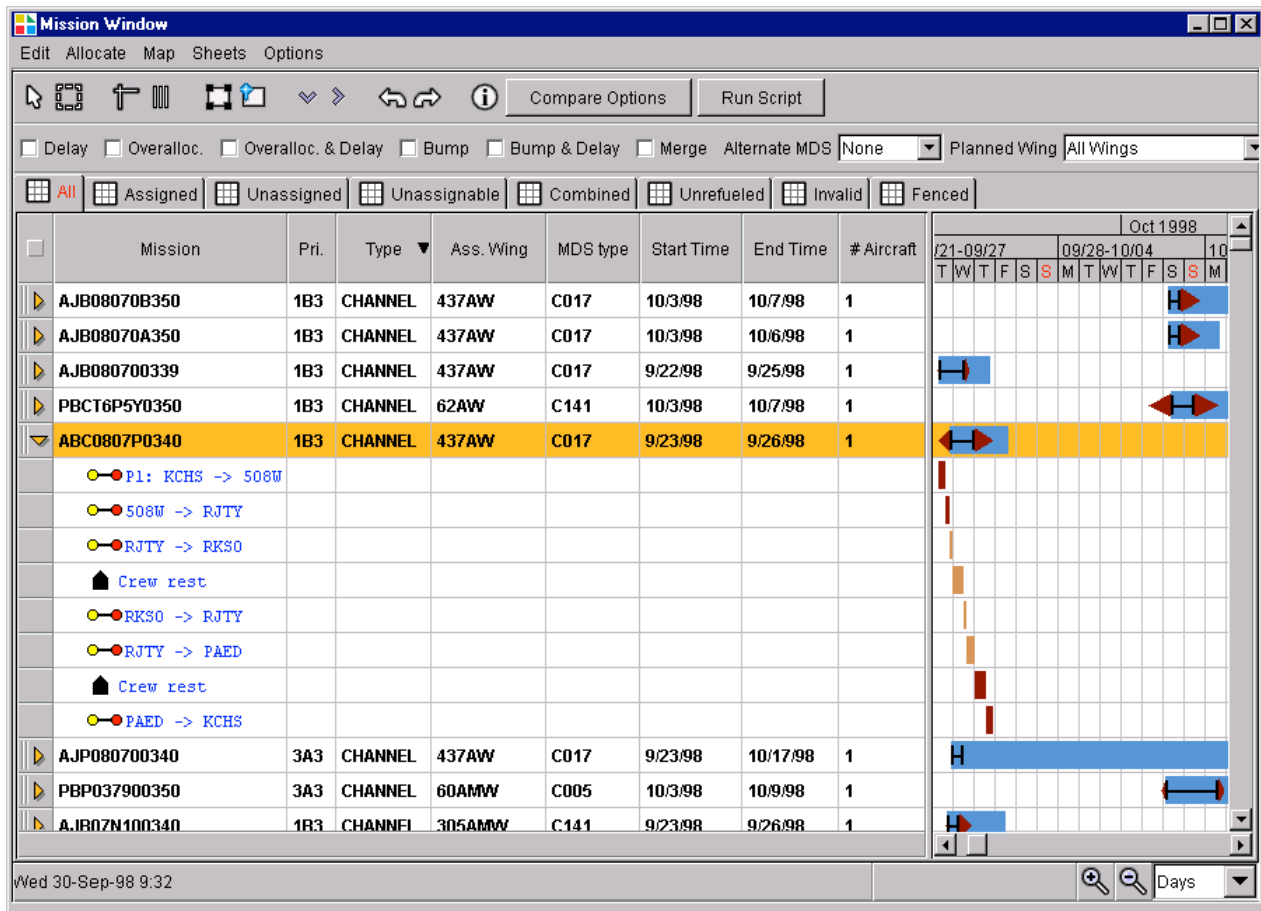


Figure 1 Barrel Allocator Mission Window

computational paradigm, and the look and feel of the core interface elements themselves. At any point in the construction of a schedule the user can rely on the system to quickly propagate effects in batch or individually from initial data to end decisions, and then be ready for the user to initiate “what-if” scenarios on unallocated missions, particularly those that are in a state of conflict. The user can UnDo and then ReDo allocations, request a system rationale for an allocation choice, and compare various allocation options.

The principal view into the Barrel Allocator is the Mission Window (Figure 1, above), organized into tabbed

The left hand side of a sheet displays the missions and their attributes. Each individual mission can be drilled down (see mission ABC0807P0340) to display individual legs (flight segments). Some of the legs are pre-planned input to the system, and others, such as positioning and de-positioning legs and crew rest, are computed and inserted by the system during the scheduling process. The right hand side of a sheet shows the mission time constraints, and where a mission schedules (the “I-beam”, with the arrows being positioning and de-positioning times) with respect to its feasible time window (the shaded area). This spreadsheet-like interface is useful because it encapsulates a good deal of information into a compact space, is very familiar to the system users, can be sorted in various ways, and allows missions to be grouped according to current state and user-defined criteria. The Mission Window, though, is only one of a number of ways that users can interact with a mission or groups of missions. Figure 2, the Mission Map Window, geographically displays the same mission that is selected in Figure 1, and Figure 3, the Capacity Window, shows a capacity timeline for the entire 437AW C017 wing, with the interval for mission ABC0807P0340 highlighted.

A Case Study

To better understand the use of the Barrel Allocator interface in mixed-initiative resource allocation, we present a brief case study of how it might be used to incorporate hurricane relief airlift missions into an existing schedule. Figure 1, The Mission Window, displays a few of close to 1000 missions that have already been allocated over a four week scheduling horizon.

A Barrel Master tasked with hurricane relief for Puerto Rico loads in 40 new missions, all of which, by default, are not allocated and thus are grouped and viewable in the Unassigned Sheet. Scheduling 40 missions is not a big task and *could* be completed manually, but it would certainly be difficult to do in anywhere near an optimal way by hand. (Moreover, in crisis situations, significantly larger sets of movement requests may need to be integrated, typically under considerable time stress.) So, the user requests the system to auto-allocate the new missions. The Barrel Allocator is able to assign all but four of the unassigned missions. These four missions are now considered

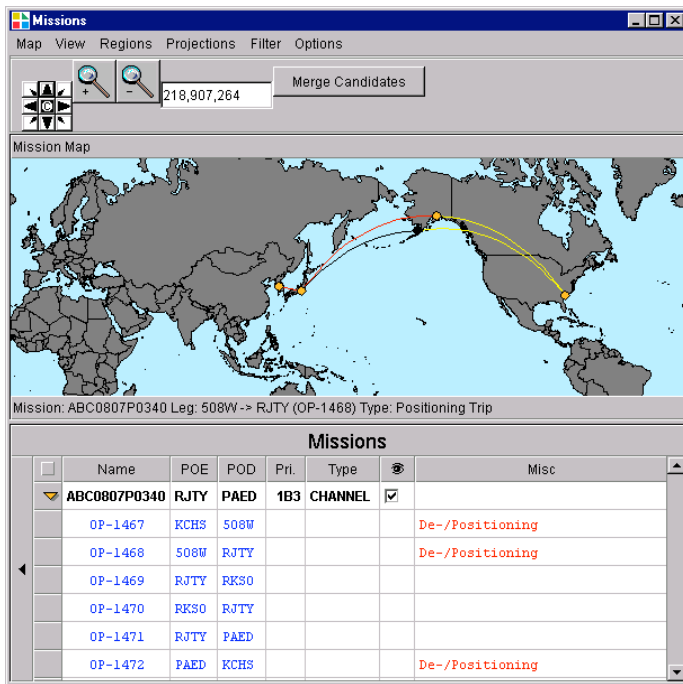


Figure 2 Mission Map Window

sheets that classify missions by their current allocation status (e.g., “Assigned”, “Unassigned”, “Unassignable”).

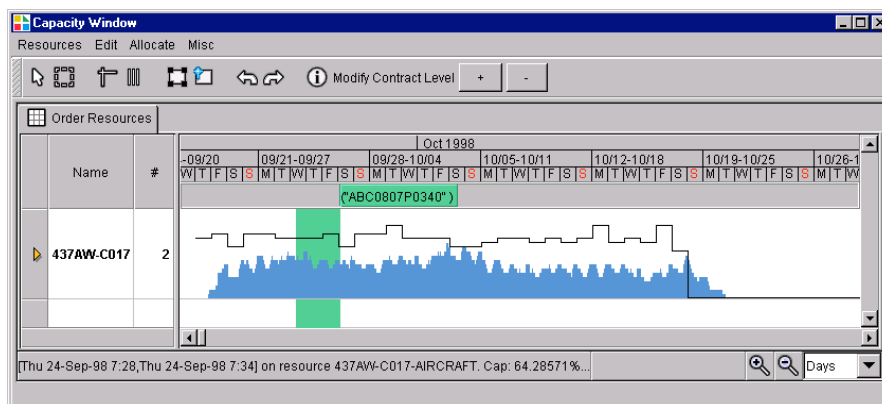


Figure 3 Capacity Window

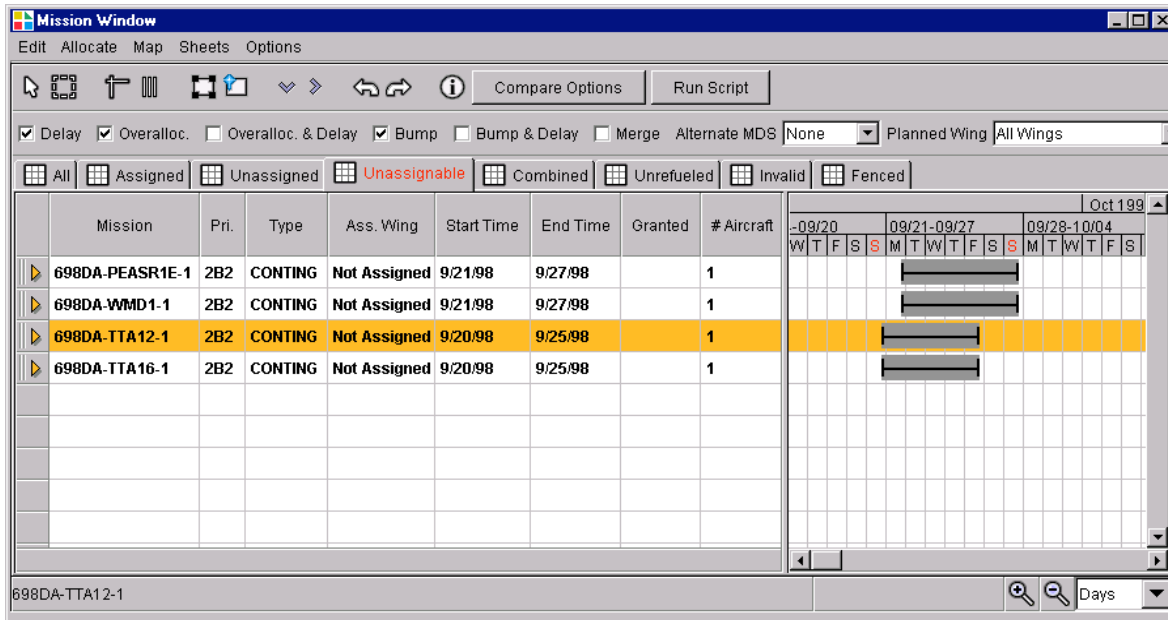


Figure 4 Unassignable Sheet

“Unassignable” as can be seen in Figure 4.

Unassignable Missions

Unassignable missions are those missions that have been considered for allocation, but have been unable to be assigned *given the constraints currently in force*. This is a very important qualification as many of the constraints in the domain that are typically in force are not truly hard constraints, but instead can be viewed as relaxable restrictions placed on the mission when it was initially planned. For instance, mission 698DA-TTA12-1 in Figure 4 was planned to utilize a C141 MDS (aircraft) type. It so happens, though, that there is no C141 wing with available

capacity during the time period for which this mission is planned. This fact is not obvious, but can be determined by querying the system “Why didn’t this mission schedule?” by requesting a “Scheduling Transcript” (see Figure 5). The transcript shows that aircraft and/or crews for all wings were lacking to support the mission.

At this point the Barrel Master could ask the system to allocate the mission to an alternate MDS type, say a C017, but suppose s/he knows that the mission in question has good reasons for flying on a C141. There are a number of other constraints that can be relaxed to attempt to get the mission on the schedule.

Comparing Options

Besides the option of allocating with an alternate MDS type, the user could explore possibilities for over-allocating one of the C141 wings. This is quite a common alternative, as a wing generally has reserve capacity that might be made available through negotiation with responsible personnel. In addition the user could consider delaying the mission, or possibly “bumping” an already scheduled mission.

Exploration of these constraint relaxation options is difficult to fully automate for a couple of reasons. First of all, it is difficult a priori to weigh the benefit of relaxation along varying dimensions of a problem space. [Smith, Lassilia, and Becker, 1996] How does delaying a mission by 17.247 hours compare with over-allocating a wing by 17.247 percent? Secondly, actually committing to one decision as opposed to others often requires a human to contact another human to negotiate agreement. In theory

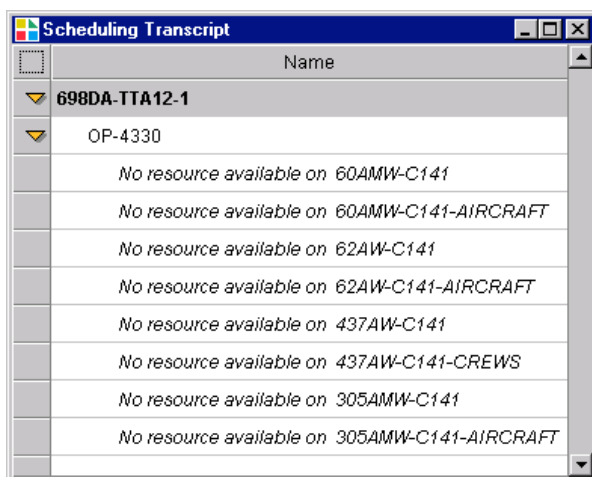


Figure 5 Allocation Transcript

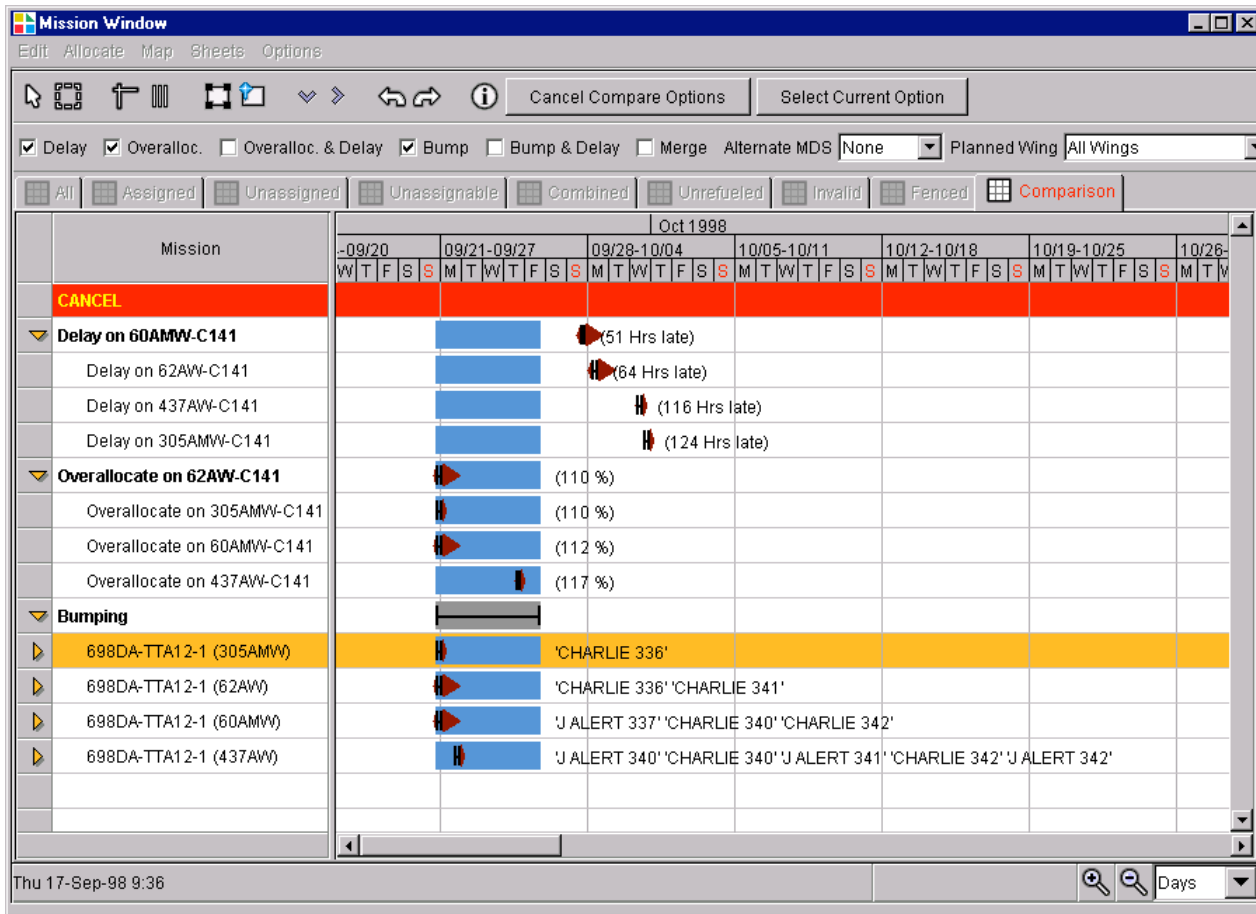


Figure 6 Compare Options Results

two intelligent, automated scheduling agents could conduct this negotiation, but in practice, that alternative is not practical for most decision making.

The Barrel Allocator provides the user with a powerful, semi-automated mechanism, “Compare Options,” which allows the user to select which constraints to relax, and then presents the user with the results – if any – of allocating a mission under various relaxation scenarios. The results are ranked within each dimension (for instance, delaying a mission by two days is ranked more highly than delaying by five), but it is up to the user to select which option to commit to. Even after the user commits to a particular decision, the decision is not final. S/he has the ability to analyze how the decision has propagated through the schedule (much like doing “What-if” in a spreadsheet), and of reversing the commitment (by pressing the UnDo button), and selecting another option for analysis or final commitment.

Returning to our “hurricane relief” case study, the Barrel Master has chosen to explore the options of delaying, over-allocating, or bumping in order to schedule mission 698DA-TTA12-1. The results that the system has generated can be seen above in Figure 6. Clearly, delaying

is not an acceptable option, as the best delay choice presented would move a hurricane relief mission back 51 hours. Other delay options presented get progressively worse. Over-allocation is a possibility, as four different options exist requiring from ten to seventeen percent allocation over contracted capacity after the mission is allocated. It’s quite possible that a wing commander could free up a fenced (reserved for local usage) aircraft to service a critical relief mission.

In this case, though, the Barrel Master might find the first bumping option the most attractive alternative. It turns out that this option involves leaving unscheduled only one lower priority mission – “CHARLIE 336.” (Other bumping options require unscheduling two or more missions.) After s/he bumps this low priority mission with the higher priority one, it is still possible to explore options for allocating CHARLIE 336 and possibly get it back on the schedule by delaying it only slightly.

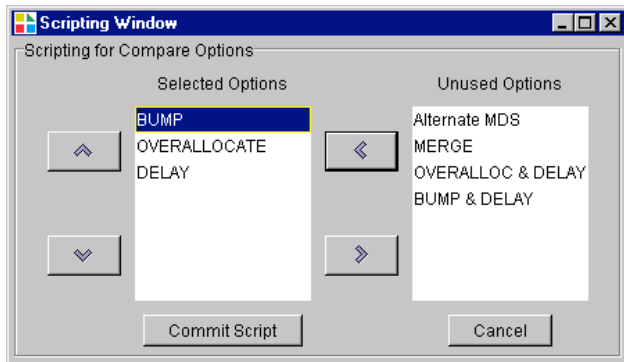
A Continuum of Automation

The brief hurricane relief case study illustrates a continuum (from partial to full) of automation available to the users of the Barrel Allocator. They can push a button

and request that the system allocate a large group of new missions into the backdrop of an existing schedule. With the aid of the system, they can examine scenarios for allocating conflicted missions and make the final allocation decision *themselves*. In addition, they can provide a little guidance to the system in the form of scripts and direct commands, but let the system make the final choice.

For instance, suppose the user were pressed for time and knew that a good way to allocate a particular mission would be to bump another one. They could do this manually by finding a suitable mission to bump, de-allocate it, and allocate the conflicted one.

Given the complexity inherent in a backdrop of a



thousand missions, doing this could be time-consuming and error-prone. A better solution is available, by requesting the system to allocate the best bumping option for the selected mission. Given our example, the system would bump the same CHARLIE 336 mission that the user settled on after reviewing a number of options.

This approach could possibly be a temporary dead end, though. Suppose no missions are available for bumping. Another semi-automated solution available to the user is to graphically construct a script (Figure 7) that guides the system in what options to try first, but to stop and return the first solution found. So, the user could indicate: first try all bumping options, then try all over-allocation options, and finally try delay options.

A continuum of automation methods makes the Barrel Allocator a very powerful tool in the hands of the end user. Some may start by using the tool in a fairly manual mode, but progress to more highly automated methods as they gain trust that the system makes reasonable choices. Users may opt for highly automated techniques when time is of the essence, and choose to do more iterative scenario exploration when more time is available, or if an automated decision is not the best one given knowledge they possess that the system does not.

Discussion: Benefits of Flexible, Mixed-Initiative Interfaces

While there is a great deal of agreement in the AI planning and scheduling community with regard to the deficiencies of black-box, fully-automated approaches to problems, there is not such unanimity in terms of what constitutes a

good design of a mixed-initiative interface to planners and schedulers.

Myers, presenting the work on the Advisable Planner [Myers, 1996], argues that users “want to be involved with the planning process at a higher, more strategic level,” leaving the low-level planning details to the system itself.

Cox and Veloso, describing the “glass-box” approach of the PRODIGY planning system argue that for some domains it is desirable to allow “the user to exert control over all aspects of decision making in the planner.” [Cox and Veloso, 1997] They go on to explain that decision making not currently under the focus of attention should be “abstracted or hidden from the user.” In other words, though the box be glass, it shouldn’t be completely transparent at all times from all angles. The benefit to be gained from this approach is that different users and a given user over time can become comfortable with the level at which they best interact with the system without being overwhelmed by irrelevant details.

This philosophy of building flexibility of interaction into the system has similarly motivated our own work with the OZONE Planning and Scheduling Framework [Smith, Lassila, and Becker, 1996] and the benefits are born out in the AMC Barrel Allocator, as we have described above. Experimental results of users interacting with a vehicle routing system to improve optimization results reported in

Figure 7 Scripting Window

[Scott, Lesh,

and Klau, 2002] further support the claims that a) human intervention can produce better results and b) different subjects interact differently with the system and perform better on different optimization sub-tasks.

Opening up the optimization process of a system to human interaction is not without pitfalls. It is often the case that human intervention in the process can produce a worse solution. This danger, though, is far outweighed by the benefit of human involvement – flexible human involvement – with real world planning and scheduling systems. Quite often the biggest impediment to the success of planning and scheduling systems is whether they get used at all. In many organizations there is understandable inertia to using new systems to improve their business processes. Users who are comfortable, even when overworked, in their current manual and homegrown approaches to resource allocation problems amplify this inertia.

It is particularly difficult to quickly change from a mainly manual process to a fully automated style of work. Encouraging users to interact with the system at different levels of automation through intelligent user interface design and intelligent algorithms, allows the users to become progressively comfortable with the system and makes them into enthusiastic collaborators. In the design of the Barrel Allocator, care was taken to provide a continuum of automation modes which, at the lower

extreme, simply mimics the prior manual process with constraint checking and option generation support.

The notion of humans and computers as collaborators in problem solving through the use of intelligent interfaces has been the focus of work on the COLLAGEN system. [Rich, Sidner, and Lesh, 2001] The authors raise a very high bar for intelligent user interfaces with the claim that every interface should have (amongst other requirements) an Undo button, to relieve the user of the fear of making a mistake, and a “What should I do next?” button to help the user figure out the next most logical task to tackle. The Barrel Allocator *has* an UnDo button, and it’s difficult to imagine the system without it. Not only does it relieve the user of fear of making a mistake, it opens up the possibility of exploring “what-if” scenarios of almost unlimited complexity.

It can be argued that the Barrel Allocator has a “What do I do next?” button as well – the “Compare Options” button. Pushing this button usually presents the user with not one, but a number of suggestions for what to do next that are detailed and finely nuanced (see Figure 6, above). Natural language dialogue, as in the TRAINS [Ferguson, et al., 1996] and TRIPS [Allen, et al., 2001] systems, where a human and computer collaborate on a planning task through dialogue, is but one mechanism amongst many for human/machine collaboration, as [Rich, Sidner, and Lesh, 2001] point out.

Are Mixed-Initiative Interfaces Always Appropriate?

We can think of planning and scheduling systems where mixed-initiative interfaces might not be appropriate. What about the system designed for a deep space probe that is to operate autonomously – out of contact with its users for long periods of time, if not indefinitely? One issue is that for the foreseeable future such systems will need to be tested and validated. Before the validators cede total control to the system, they’ll need to interact with it, and that interaction can best be played out along a continuum of automation – at times leaving the system alone, and at other times pestering it as to why it made this decision as opposed to that one.

The Barrel Allocator: Future Work

The Barrel Allocator is in its infancy of operational use, and demands from users of the system will certainly drive much of the future work and research direction. That having been said, there are a number of areas of future work we have identified and some that we’ve begun exploring.

A focus of this paper was on the semi-automated aspects of the optimization process. Complementary to this is work on improving the core, automated search facility. One area of active interest is extension to allow the system to search more exhaustively, but under time-bound controls that the user has set. This would allow for “comparing options” at a higher level of abstraction: comparing one schedule versus

another, as opposed to just individual mission allocation. Evolving good metrics for schedule comparison is in itself an interesting area for study.

Another area of work that is already in progress is improving the Barrel Allocator’s “Explanation” facilities, particularly in the case of failure to allocate. The case in Figure 5 illustrates a simple failure to find resource capacity from several wings. Failure to allocate an air refueling request quite often involves more complex causes, which need to be presented to the end user in a clear and concise way. Figure 8 represents a transcript of a failure to schedule an air refueling request as it is currently displayed. Perhaps a natural language summary of the information presented could remove some of the redundancy and make it easier to digest.

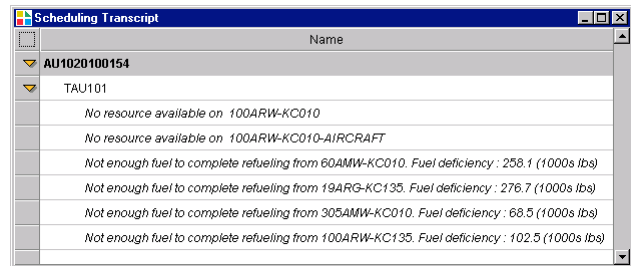


Figure 8 Scheduling Transcript for an Air Refueling

Finally, we are beginning to look into ways that the scheduler can be even more responsive to change due to access to real-time, “state of the world” inputs. Currently the system does a good job of automatically and semi-automatically incorporating new missions into an existing schedule. What we would like to be able to handle in a graceful and intuitive way is feedback into the schedule such as a particular destination airbase has been put off limits for a period of time, or that a particular mission was delayed well beyond its scheduled departure time.

Acknowledgements

We thank Marcel Becker for the years of work he contributed to help make the Barrel Allocator possible. This work has been funded in part by the Department of Defense Advanced Research Projects Agency and the US Air Force Rome Research Laboratory under contract F30602-97-2-0227, by the USAF Air Mobility Command under subcontract 10382000 to Northrop Grumman Information Technologies, and by the CMU Robotics Institute.

References

Allen, J., Byron, D.; Dzikovska, M.; Ferguson, G.; Galescu, L.; and Stent, A. 2001. Toward Conversational Human-Computer Interaction. *AI Magazine* 22(4)27-37.

Becker, M.A. and Smith, S.F. 2000. Mixed-Initiative Resource Management: The AMC Barrel Allocator. In *Proceedings 5th International Conference on Artificial Intelligence Planning and Scheduling (AIPS-2000)*, Breckenridge, CO, April 2000.

Burstein, M. and McDermott, D. 1994. Mixed-Initiative Military Planning: Directions for Future Research and Development. In *Proceedings ARPA/Rome Labs Planning Workshop '94*, Tucson AZ, February 1994.

Cox, M.T. and Veloso, M.M. 1997. Supporting Combined Human and Machine Planning: An Interface for Planning by Analogical Reasoning, In *Case-Based Reasoning Research and Development, Proceedings of ICCBR-97, the Second International Conference on Case-Based Reasoning*, 531-540. Berlin: Springer Verlag.

Ferguson, G., Allen, J. and Miller, B. 1996. TRAINS-95: Towards a Mixed-Initiative Planning Assistant, In *Proceedings Third Conference on Artificial Intelligence Planning Systems (AIPS-96)*, Edinburgh, Scotland, May 1996.

Myers, K. L. 1996. Advisable Planning Systems. In A. Tate, editor, *Advanced Planning Technology*. AAAI Press, Menlo Park, CA.

Rich, C.; Sidner, C.; and Lesh, N. 2001. COLLAGEN: Applying Collaborative Discourse Theory to Human-Computer Interaction. *AI Magazine* 22(4), 15-25.

Scott, S.; Lesh, N.; and Klau, G. 2002. Investigating Human-Computer Optimization. In *Proceedings of CHI 2002 (Conference on Human Factors in Computer Systems)*, Minneapolis, MN, April 2002.

Smith, S.F.; Lassila, O.; and Becker, M.A. 1996. Configurable, Mixed-Initiative Systems for Planning and Scheduling. In A. Tate, editor, *Advanced Planning Technology*. AAAI Press, Menlo Park, CA.