Mixed-Initiative Resource Management: The AMC Barrel Allocator

Marcel A. Becker and **Stephen F. Smith***
The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue Pittsburgh, PA 15213
mb81,sfs@cs.cmu.edu
(412) 268-8811

Abstract

In this paper, we describe the *Barrel Allocator,* a scheduling tool developed for day-to-day allocation and management of airlift and tanker resources at the USAF Air Mobility Command (AMC). The system utilizes an incremental and configurable constraint-based search framework to provide a range of automated and semi-automated scheduling capabilities, including generating an initial solution to the fleet assignment problem, selective re-optimization of resource allocations to incorporate new higher priority missions while minimizing solution change, merging of previously planned missions to reduce non-productive flying time, and generation and synchronization of tanker missions to satisfy air refueling requirements. In situations where all mission requirements cannot be met, the system can generate and compare alternative constraint relaxation options. The current version of Barrel Allocator will go into operational use at AMC as a module of Release 2.0 of AMC's Consolidated Air Mobility Planning System (CAMPS) in early 2000.

Introduction

Efficient allocation of aircraft and crews to transportation missions is an important priority at the USAF Air Mobility Command (AMC), where airlift demand must increasingly be met with less capacity and at lower cost. The AMC resource management problem presents several interesting challenges:

- problem scale - Over a typical short-term (e.g., 2 week) scheduling horizon, several thousand air missions are flown world-wide, utilizing several hundred aircraft and active-duty air crews. In times of crisis, these numbers can increase substantially, and additionally involve both reserve units and commercial aircraft.
- problem complexity - Resources must be allocated to missions in a way that minimizes non-productive flying time, attends to mission priorities, and maximizes the number of supported missions, while at the same time ensuring that decisions are feasible with respect to aircraft availability and operating characteristics, crew duty day limits, required mission execution windows, airport capacity and landing time restrictions, and other mission constraints. In cases where all constraints cannot be satisfied, some may be selectively relaxed and tradeoffs must be made between alternative options.
- solution continuity - Like most practical domains, resource allocation is situated in a continuously executing environment. Each time a resource assignment is changed, new orders must be re-communicated to the affected wings and re-assimilated into locally planned activities. Hence, it is important to manage and minimize solution change as new missions are incorporated into the schedule over time.
- interacting planning and scheduling problems - Effective allocation of resources requires a tight interplay of planning and scheduling capabilities. By default, missions are planned as round trip operations and each assigned aircraft returns to its home base upon completion. However, lift capacity can often be increased by combining two or more missions and "recycling" the same aircraft from one mission to the next. Alternatively, a planned airlift mission may require air refueling. In this case, tanker capacity must be sourced, and a supporting tanker mission must be generated and synchronized in time.

Due to the time pressure of decision-making and the lack of automated scheduling tools, the AMC "Barrel Masters" responsible for making allocation decisions typically make allocation decisions in a limited, myopic fashion and routinely miss opportunities to optimize resource usage.

In this paper we describe Barrel Allocator, a tool for generating and evaluating such optimization opportunities. Developed through application of the Ozone scheduling framework (Smith *et al.* 1996; Becker 1998), Barrel Allocator utilizes incremental, constraint-based scheduling techniques to allow integration of new missions and response to changing requirements and availability, while minimizing disruption to most previous

*Authors listed in alphabetic order

assignments. In situations where all constraints cannot be satisfied, the Barrel Allocator's search model promotes selective constraint relaxation to find an acceptable solution. Mission scheduling and resource allocation capabilities can be invoked in automated or semi-automated modes; in the latter case, the system generates and compares different options that might be taken by the user.

Experimental results with Barrel Allocator, obtained using historical data extracted from the current ADANS airlift planning system indicate the potential for substantial improvement in resource usage, through better optimization of air wing assignments, selective combination of missions to efficiently "recycle" aircraft, and more effective integration of tanker and airlift missions. Following positive review by AMC personnel, a version of Barrel Allocator has been delivered to the Tanker Airlift Command Center (TACC) at AMC for extended user review and testing. Current plans call for Barrel Allocator to go into operational use within the TACC early next year as part of release 2.0 of AMC's Consolidated Air Mobility Planning System (CAMPS).

The remainder of this paper is organized as follows. We first briefly summarize the Barrel Master allocation problem and current practice at AMC. This is followed by an overview of the functional capabilities provided by the Barrel Allocator tool. In the next two sections, we discuss the underlying representations and search procedures that provide the system's technical basis. We then provide some details on the status of the implementation and technology transition effort. Finally, we discuss some planned extensions.

The AMC Barrel Master Allocation Problem

The AMC "Barrel Master" (or Barrel for short) is in charge of resource allocation and resource management for the USAF Air Mobility Command. The different planning offices at AMC submit resource allocation requests to the Barrel in the form of *missions* or *mission requests.* These missions represent, for example, requests to move cargo and/or personnel, or requests to reserve resources for a number of training activities and exercises. Although different types of missions create different types of resource requirements, all planned missions specify a particular type of aircraft to be used, an itinerary, a priority, a preferred air force unit or *wing* to fly the mission, and a time period, represented as a set of dates, in which the mission should be executed.

Each Barrel manages particular sets of aircraft and corresponding crews. These sets are defined by aircraft type and by the geographic locations where the aircraft are stationed. A set of aircraft of the same type stationed at a particular air force base constitute a *wing.* For example, McGuire Air Force base has a wing of C141s and a wing of KC135s. Currently, one Barrel is responsible for all west coast C141s and another Barrel is responsible for all east coast C141s. Planners generally specify which wing they would prefer to fly a given mission, and when possible, the Barrel will allocate a plane from this planned wing.

The mission itinerary is the sequence of stops or airports the aircraft should visit during the execution of the mission. We refer to the flight between two successive stops as a *mission leg* or just a *leg.* Each leg has an origin airport, the Point of Embarkation (POE), and a destination airport, the Point of Debarkation (POD). Each leg is followed (or preceded) by a certain ground time. During the period the aircraft is on the ground, a number of activities, or *ground events,* can occur: for example, loading and offloading of cargo, refueling, crew rest, crew change. The time period specified in the mission request should be at least as large as the time required by the aircraft to fly between all intermediate stops in the itinerary plus the required ground time at each airport. The Barrel will try to assign one aircraft that is available during this entire period. The earliest date the mission can start is called the *Available to Load Date* (ALD) and the latest date the mission should finish is called the *Latest Arrival Date* (LAD). The length of this interval should be at least as large as the total duration of the mission.

Aircraft availability is defined for each wing on a daily basis. Each wing has a total number of aircraft of a particular type. Considering that some planes are undergoing maintenance and the wing has some need for training and local missions, the wing will make a subset of all its planes available to AMC missions. Each day, each wing will provide a certain number of *contract* aircraft that can be allocated by the Barrel. The remaining aircraft, designated as *fenced* aircraft, are reserved for local wing use and are beyond the jurisdiction of the Barrel.

In her/his daily activities, the Barrel Master currently manages resource availability using a *commitment matrix.* This matrix tracks available aircraft capacity of different wings over time and records those missions already allocated. As new missions are received from various planning offices, the Barrel consults this matrix and tries to allocate resources which satisfy mission requirements. If all requirements can be satisfied, s/he makes the aircraft assignment and communicates mission commitment back to the planner. In those cases where there are insufficient resources available to support a particular set of missions, s/he will consider more disruptive allocation alternatives. For example, s/he will consider using resources already allocated to lower priority missions or will consider using resources provided by a different wing. Once one or more acceptable options are found, the Barrel communicates these possibilities back to the relevant planner and a solution that would best satisfy all sides is negotiated.

The AMC Barrel Master problem is similar to the problem known in the OR literature as the *Fleet Assignment Problem:* Given a schedule of flights defining the departure and arrival times for each fly leg, the *Fleet Assignment Problem* is the problem of deciding which flight equipment, or fleet, should be assigned to each flight segment (Barnhart *et al.* 1998; L.W. *et al.* 1996; Rushmeier and Knotogiorgis 1997). The AMC Barrel Master problem addressed by the Barrel Allocator is a dynamic, string-based version of this problem. For commercial airlines, the objective is to maximize revenues minus operating costs. The Barrel Master tries to maximize the total sum of priorities: s/he will try to assign the maximum number of high priority missions, and would only consider assigning lower priority missions after all higher priority ones have been assigned.

Traditional OR-based solutions to the fleet assignment problem assign fleets to individual flight segments. The Barrel, alternatively, is concerned with assigning fleets to a sequence of segments or *strings.* A *string* is a sequence of connected flight segments that begins and ends at possibly different maintenance stations (Barnhart *et al.* 1998). An AMC mission itinerary is typically planned as a string that starts and ends at the same location (i.e., a round trip). Strings that start and end at the same station are usually referred as *aircraft rotations.* If possible, the Barrel would consider, and sometimes even prefer, using the same rotation for more than one mission. The difficulty in combining missions is in identifying the opportunities for potential combinations among thousands of missions in the database.

The Barrel Master in charge of refueling resources, the Tanker Barrel, in addition to allocating tankers to planned air refueling missions, is also responsible for linking *air refueling events* with regular airlift missions. An air refueling event is a request for a refueling mission that is generated each time a planned airlift mission requires air refueling. In the data repositories currently available, there is no explicit linkage between air refueling events and the missions they are supposed to serve. Thus, the Tanker Barrel currently has to perform this linkage by manually searching the database for airlift missions that matches the location and time of air refueling events.

The allocation process described in previous paragraphs is currently mostly manual. The Barrel Master uses a system that provides an electronic commitment matrix and is linked to current aircraft availability and mission data. However, resource assignment is performed one mission at a time with little automation. Capabilities for identifying mission combination opportunities are quite limited and as just mentioned there is no system support for establishing linkages between tanker and airlift missions. The Barrel Allocator system described below aims at automating some of these tasks and enhancing the decision-making capabilities of the Barrel master, while still granting full control and visibility over the decision making process.

Functional Capabilities

The Barrel Allocator provides three core sets of functionality to the AMC Barrel Master: *resource allocation, mission combination,* and *air refueling linkage.* In all cases, functionality can be utilized in a more or less automated fashion, ranging from a fully manual mode where the system does little more than decision bookkeeping, to a semi-automatic mode, where the system generates alternative options and previews their impact, to a completely automatic mode, where the system determines selects the best decisions based on user-specified preferences. In the paragraphs below, we first summarize these core functional capabilities. In subsequent sections, we then discuss their technical basis.

Resource Allocation

In a typical mode of operation the problem is one of integrating sets of newly planned missions into an existing current global schedule. In this mode, the scheduler will attempt to assign aircraft and schedule new missions without disrupting the current set of assignments. Any mission that cannot be integrated into the schedule in this way (which implies that there is not enough lift capacity to accomplish the mission without changing existing resource assignments) is flagged as *unassignable* and will require subsequent user attention.

The assignment of wings to new missions can be performed in manual or automatic mode. The user selects some set of unassigned missions, the alternative wings to be considered for those missions, and the system will compute all feasible allocations. In manual mode, the application displays all feasible solutions and the user can select the preferred one. In automatic mode the system selects the best allocation based on some user defined preference, (for example, try to use the wing the minimizes overall mission time weighted by priority). The allocation process produces an assignment of (notional) aircraft from specific wings to each mission, and an assignment of flight times to each mission leg. If the locations corresponding to origin and destination of the mission are different from the location of the wing providing the plane, positioning and de-positioning flights will be added to the itinerary.

Currently, the following constraints are taken into account and enforced in constructing a schedule for all current missions:

Wing capacity constraints - Assignment of missions to wings does not exceed the number of contract aircraft available at each wing.

Mission time requirements - Missions must be scheduled within time windows designated by ALD and LAD constraints.

Enforcement of required ground time - The allocated time accounts for aircraft onload, offload and minimum time-on-ground constraints, each specified as a function of aircraft type. More generally, it is possible to specify a range of different flight preparation activities and time constraints.

Flight duration constraints - flight duration can be specified as an input data or is computed using great circle route

Aircraft range constraints - which are enforced when determining specific wing assignments (and hence when creating positioning/de-positioning flights)

Crew duty day constraints - Depending on designated crew type - basic or augmented - crew rest is inserted at appropriate intermediate points of the mission to enforce crew duty day.

As suggested above, it is not always possible to satisfy mission time constraints with existing available lift capacity. To support resolution of such situations, the system gives the user the ability to analyze and compare various constraint relaxation options. Specifically, for any given set of unassignable missions, the user can choose to:

Allow bumping of lower priority missions - Every mission has a pre-defined priority. If there is no aircraft available to support a high priority mission, one alternative is to pre-empt lower priority missions already in the schedule. Any pre-empted missions are then rescheduled in succession and they may, in turn, pre-empt missions of lower priority still. At quiescence, any lower priority missions that cannot be be re-inserted into the schedule within its constraints are added to the unassignable list. A *mission locking* mechanism is provided to allow the user to avoid bumping any specific, lower priority mission. A *freeze interval*, a period of time in which no mission can be bumped, is also enforced. The freeze interval is required to avoid schedule turbulence close to execution.

Over-allocate - Since the number of contract aircraft is usually smaller than the total number of possessed aircraft, the user may alternatively choose to go over the published contract level of a given wing. This happens with a fair amount of frequency. It typically reflects extra knowledge that the barrel master may have about wing assets or agreement on the part of the wing to use fenced aircraft.

Delay the mission - The user may consider the option of delaying the current mission until necessary resources are available. If delay seems like a potentially viable alternative then this information can be suggested to the mission planner. Mechanisms for limiting the amount of delay acceptable and combinations of delay and bumping, and delay and over-allocation are also supported.

Use alternative MDS - Similarly, it might be possible to accommodate the mission if an alternative airframe type can be utilized.

Any of these options can be invoked by the user in "what-if" mode; a general "undo" capability allows the retraction of any sequence of scheduling actions that have been issued by the user, and thus provide a basis for exploring alternatives. Alternatively, the user can ask the system to generate all options and compare alternatives. This range of scheduling modes provides a continuum of scheduling actions that are progressively more disruptive in the changes that can be made to the current schedule. There is no formal metrics to compare alternative relaxation options. Each option will have its own disruption metric (e.g., amount of resource overallocation, number of hours late, number of lower priority missions bumped). The user is currently responsible for evaluating and selecting the best alternative based on his/her own subjective criteria.

Mission Combination

Missions are planned by default as round trips from a particular home base. If the origin and/or destination of the mission does not coincide with this base, positioning and depositioning flights segments are added to the mission itinerary. To improve resource utilization and increase aircraft availability over time, the Barrel Allocator provides the capability of exploiting mission combinations: it will look for opportunities to link two missions such that after the end of one mission, the airplane will be "recycled" and redirected to support another mission instead of depositioning back to its home base.

The mission combination capability can be used as an alternative allocation policy to allocate unassignable missions or as a compression mechanism to increase the number of airplanes available over a certain period of time. The user can select or filter the potential merge candidates using three different parameters:

Maximum layover time - the maximum time delay that can be tolerated between offload (end) of one mission and onload (beginning) of second mission

Maximum distance - the maximum distance that can be tolerated between location of first missions's offload and second's onload

Percentage decrease in overall flying time - the reduction obtained by combining two missions into one aircraft rotation compared to flying both original round trips.

Air Refueling Linkage

To support the additional responsibilities of the Tanker Barrel, capabilities are also provided for generating and assigning tanker missions to airlift missions that require air refueling support. A set of refueling events is accepted as input, and tanker assignments can be generated either interactively or automatically. In automatic mode, the scheduler first attempts to link tanker missions that are already included in the schedule (e.g., training missions); tanker wing assignments are determined and tanker missions are created for any remaining unsupported airlift missions.

In interactive mode, a map-based display is used to indicate candidate refueling tracks that are already covered by tanker missions in the temporal interval required by a given airlift mission. The user can select one of these highlighted refueling tracks (potentially changing the originally planned refueling location) or select the planned track (which will result in the creation of a new tanker mission if one is not already in the schedule). In the case of the selection of a pre-existing tanker mission, the system will also present opportunities to support multiple air refueling events with the same tanker mission (provided that fuel requirements and tanker fuel capacity constraints are satisfied).

Problem Formulation and Representation

As indicated at the outset, the Barrel Allocator has been developed through use of the modeling primitives and class library defined in the Ozone scheduling framework. Development of an application system in Ozone involves three basic tasks:

1. Identifying the ontological entities in the domain relevant to the application domain,
2. Defining and developing a domain model, by mapping the problem specific ontology onto the Ozone scheduling ontology,
3. Instantiating and/or extending Ozone problem solving templates to address the set of relevant constraints.

In this section we describe the Barrel Allocator's domain model. In the next, we consider its problem solving procedures.

Ozone domain models are defined in terms of five basic entities (Smith and Becker 1997): *demands, activities, products, resources,* and *constraints.* The `mission requests` sent by the planners to the Barrel correspond to the demands. They are used to translate the external requirements specified by the user into the internal constraint model used by the scheduler. The basic attributes of the mission request are: a pair of dates establishing the earliest date the mission should start and the latest date it should finish; the type of aircraft and preferred wing providing it; the mission priority (which imposes a partial order on the set of missions); the itinerary or set of flight segments the mission should fly; and the mission type. The `mission type` is the product (i.e., service) of interest. Satisfaction of a mission request implies that a certain type of service has been provided, and the product is the entity that represents this service within Ozone models. Services are provided through the execution of activities. The `flight segments`, or `legs`, and `ground events` are the principal types of activities in this domain. Each type of mission imposes special constraints on how the activities should be performed. For example, refueling missions and fuel supply missions require special types of resources and temporal synchronization between sets of activities. Two special types of flight segments are the `refueling leg` and the `fuel supply leg`. The refueling leg corresponds to the activity of receiving fuel, and the fuel supply corresponds to one of providing fuel. Such problem-specific knowledge about the constraints associated with different mission types is encoded into the product entity and used during instantiation and scheduling of mission activities.

A `flight segment` activity requires four different types of resources: the `aircraft`, the two `airports` corresponding to origin and destination of the flight, and the `air crew.` Associated with each resource is the notion of capacity: the amount or quantity of a certain type of unit that is available over time. Aircraft availability is defined at the aggregate fleet level. A fleet or `wing` is a pool of aircraft of the same type located at a certain geographic location. Each aircraft in a wing represents one unit of capacity. The total number of aircraft belonging to a given wing defines its `total capacity`. Since each plane can only fly one mission at a time, the total capacity is the maximum number of missions that can fly simultaneously at any given time. The total capacity of the wing is divided into two subsets: the `contract capacity`, representing the set of planes available to support AMC missions, and the `fenced capacity`, representing the set of planes reserved for local missions and cannot be used by the Barrel in normal circumstances. We only check airports for airplane compatibility restrictions. Support for air crews is not currently supported and will be included in the next version of the system.

Resource capacity is defined over `capacity intervals.` Each interval has start and end times, and values for total, contract, fenced, and currently available capacity. The end time is always larger than the start time and the minimum size of an interval is one time unit. Time is represented continuously to the granularity of the time unit. For example, the Barrel Allocator uses minutes as the time granularity. Therefore, the minimum distance between two time points is one minute.

The output of the Barrel Allocator is a set of `resource assignments` or `resource reservations`: tuples of the form `(Mission, Wing, start-time, end-time).` This means that mission `Mission` is reserving one unit of capacity of resource `Wing` during the time interval starting at `start-time` and finishing at `end-time`. The duration in minutes of this interval – the distance between start and end times, is equal to the total mission duration. This duration includes the duration of all the flight segments, plus required time for all ground events. As we will discuss in the next section, different constraints may be satisfied by this assignment, depending on the set of parameters selected by the user while activating a particular problem solving method.

Problem Solving Methods

The planning and scheduling procedures used to provide various functional capabilities within the Barrel Allocator are instantiations of basic search method templates available in the Ozone scheduling framework. In

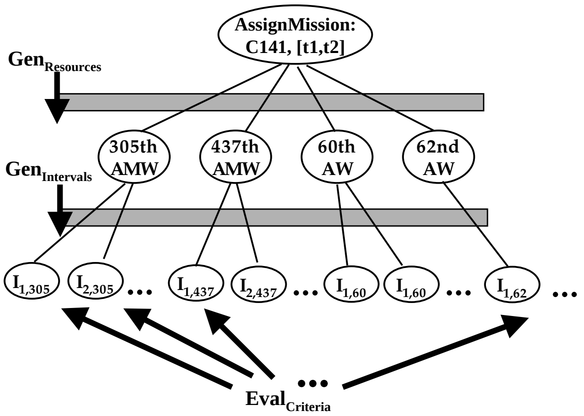

Figure 1: Basic search procedure for resource allocation (assuming that mission being assigned requires $C141$ aircraft with earliest pickup at $t1$ and latest delivery by $t2$).

Ozone, search templates are only partially instantiated, producing parameterizable problem solving procedures. These procedures can be dynamically configured to provide different search behavior as the problem solving situation dictates. In the subsections below we describe these core procedures and how they are composed to produce user level functionality.

Resource Allocation

The central function supported by the Barrel Allocator is that of assigning planned missions to wings over time. Figure 1 graphically depicts the general search template used to make this assignment for a single mission.

The search procedure proceeds in 3 steps:

1. a set of candidate resources (wings) is generated,
2. for each candidate wing, a set of possible allocation intervals is generated, and
3. each <`wing, allocation interval`> pair is evaluated and the highest ranked candidate is selected.

As implied by Figure 1, a full instantiation of the `AssignMission` search procedure is obtained by specifying three components: a search operator for generating candidate resources (referred to generically as $Gen_{Resources}$), a search operator for generating candidate allocation intervals (generically called $Gen_{Intervals}$), and an evaluation metric ($Eval_{Criterion}$) for ranking alternatives. By parameterizing the procedure to operate with different sets of operators and evaluation criteria, resource assignments can be generated and evaluated under a range of different constraint relaxation assumptions.

In the most basic case, `AssignMission` is configured to search only for feasible assignments, i.e., <`wing, allocation interval`> pairs that are consistent with the time and resource requirements specified by the mission and are also compatible with the assignments

```
AssignMissions{Unassigned_Missions, config}
   While Unassigned_Missions is not empty
   Do
        Extract a mission M from Unassigned_Missions;
        If AssignMission(M, config)
             Then Mark M as assigned
             Else Mark M as unassignable;
   EndWhile
End
```

Figure 2: Overall Mission Scheduling Procedure

of previously scheduled missions. This *feasible* configuration of `AssignMission` is obtained by incorporation of the triple $< Gen_{RequestedRes}$, $Gen_{FeasibleInts}$, $Eval_{MinFlyingTime}>$. Here, $Gen_{RequestedRes}$ generates candidate wings consistent with aircraft type requested by the mission. Likewise, $Gen_{FeasibleInts}$ scans a candidate wing's capacity profile for allocation intervals (1) with at least one unit of available capacity (i.e., an aircraft), (2) with a duration greater than or equal to the time required to accomplish the mission (a function of mission itinerary, aircraft speed, wing's home base location, crew rest requirements, etc.), and (3) with start and end times that satisfies the mission's earliest onload time and latest off-load time constraints. Candidates are differentiated on the basis of total flying time and the candidate assignment that minimizes this metric is selected.

By selectively substituting different search operators and/or evaluation criteria, `AssignMission` can alternatively be used to find assignments under various relaxed problem assumptions. For some types of constraints, relaxation simply implies the consideration of a different discrete set of options. For example, substitution of $Gen_{AlternativeRes}$ for $Gen_{RequestedRes}$ results in generation of assignments that consider types of aircraft other than the type requested by the mission planner. For other classes of constraints, however, relaxation is more continuous in nature, and in substituting a search operator that assumes constraints can be relaxed, the search must also be biased to promote their satisfaction to the extent possible.

By varying the operator used to generate allocation intervals and the evaluation metric used to prioritize candidate solutions, a number of useful `AssignMission` configurations are defined:

Delay - Incorporation of the triple $<Gen_{RequestedRes}$, $Gen_{DelayInts}$, $Eval_{MinTardiness}>$ yields an assignment procedure which assumes that mission deadlines can be relaxed if necessary. $Gen_{DelayInts}$ uses the same mechanism used by $Gen_{FeasibleInts}$ but considers a larger portion of the candidate wing's capacity profile, and $Eval_{MinTardiness}$ ensures that the mission deadline will be relaxed to the minimum extent possible

In this configuration advantage can be taken of the relationship between search operator and evaluation criterion to effectively constrain the number of candidate solutions generated. For example, by scanning forward in time through a resource's capacity profile, the first interval with available capacity found will be the one that minimizes delay for that resource. If this approach is taken only one interval need be generated for each candidate resource. In other configurations (e.g., the *pre-emption* case below), where there is no such dominance condition for constraining solution generation, more ad-hoc heuristic cutoffs can be used.

Over-allocate - The triple $<Gen_{RequestedRes}$, $Gen_{OverInts}$, $Eval_{MinOverUsage}>$ defines an assignment procedure where capacity constraints are relaxable. $Gen_{OverInts}$ scans the capacity profile of a candidate wing, but generates allocation intervals that extend above the wing's "contracted" level (i.e., dipping into its locally reserved or "fenced" pool of aircraft capacity). $Eval_{MinOverUsage}$ promotes selection of the generated allocation interval that minimizes the level of over-allocation.

In this case, maximal intervals at different levels of over-allocation can be efficiently generated via a linear scan of the resource's capacity profile, and subsequently pruned to minimize the temporal extent of over-allocation.

Priority-based Pre-emption - A configuration which assumes that some number of previously made assignments can be relaxed (or disrupted) is defined by the triple $<Gen_{RequestedRes}$, $Gen_{BumpInts}$, $Eval_{MinDisruption}>$. This configuration implements a form of pre-emption, based on mission priority. In scanning a candidate wing's capacity profile, $Gen_{BumpInts}$ considers capacity currently allocated to lower priority missions as available for assignment, and generates allocation intervals based on this assumption. $Eval_{MinDisruption}$ promotes allocation intervals that disrupt the fewest missions and those with the lowest priority. This minimizes the cascading effect (since any mission that is pre-empted by a higher priority mission is recursively re-scheduled using the same procedure).

Given the combinatorial number of allocation intervals and possible sets of bumped missions that can be generated via a complete capacity profile scanning procedure ($O(c^f)$ where c is the capacity of the resource and f is the duration of capacity profile fragments), our current implementation utilizes a linear, heuristic sampling strategy ($O(cf)$) compatible with $Eval_{MinDisruption}$. Briefly, allocation intervals are generated by single forward scan through the resource's capacity profile over the time interval where capacity is required. At each time point encountered during the scan, the set of pre-emptable missions is computed. If non-empty, the current allocation interval is extended by (1) computing the subset of preemptable missions of lowest priority and (2) selecting the mission in this subset with maximal finish time.

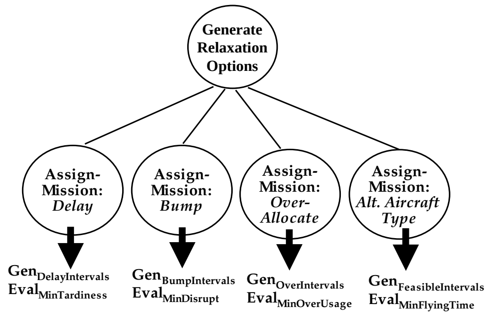

Figure 3: Exploration of constraint relaxation options using configurable `AssignMission` procedure.

Composite Relaxations - Components of the above base configurations can also be composed to define configurations of `AssignMission` where multiple constraints are simultaneously relaxed.

`AssignMission` (in any of the configurations described above) can be applied to any selected set of missions via the `AssignMissions` procedure given in Figure 2. Within this CSP-style search procedure, the current implementation uses mission priority and latest delivery date as a heuristic basis for mission selection (i.e., variable ordering). Input missions are sorted on this basis and `AssignMission` is then sequentially applied to each to allocate required resources (using whichever of the above configurations as been designated).

In cases where a given mission has no feasible assignments (i.e., `AssignMission(M, feasible)` is applied and returns no solution), an exploration of possible relaxation options can be conducted through repeated application of `AssignMission` in different configurations (as depicted in Figure 3). Automation of this process requires a global evaluation criterion suitable for relating options generated across different invocations of `AssignMission` (e.g., some measure of overall benefit and cost). In the current implementation, this procedure is used in what-if mode to generate alternative options, and the user performs the evaluation and selection.

Mission Combination

A second core function provided by the Barrel Allocator is mission combination, a capability quite analagous to the merging of similar jobs in a job shop to reduce setup costs and increase resource availability. The implementation of this capability extends the basic template presented in the previous paragraphs, coupling the execution of a "planning" component that computes alternative composite mission itineraries with a "scheduling" component that verifies feasibility of a given combined mission and generates a wing assignment. The scheduling component is the basic `AssignMission` already defined.

More generally, mission combination can be seen as a specific configuration of an integrated planning and scheduling search template, designated by the following triple of components: $<Gen_{PossComb}, Gen_{FeasComb}, Eval_{MaxTripRed}>$ where $Gen_{FeasComb}$ is an instantiation of `AssignMission`. In this case, the operator $Gen_{PossComb}$ represents the planning component. It will generate a set containing all possible composite missions involving a certain mission M. For any pair of missions (M_1,M_2), there are two possible combinations: one in which mission M_2 flies after the end of mission M_1's last leg; and one in which mission M_1 flies after mission M_2. Possible combinations must satisfy the conjunction of constraints established by spatial and temporal restrictions, plus constraints following from the three external parameters specified by the user: minimal percentage reduction in total duration of the combined mission, the maximum layover time, and the maximum distance between the end location of the first mission and the start of the second. The output of this operator can be seen as the set of all notional combined missions $\overline{M}$ resulting from concatenation of possible combination pairs (M_1,M_2).

Once all the possible combinations has been generated, the existence of a feasible wing assignment will be determined via application of `AssignMission(`$\overline{M}$`,feasible)`, where $\overline{M}$ represents the mission generated by combining the itinerary of two missions in a possible pair (including an intermediate connecting flight leg if necessary). The allocation interval should be large enough to accommodate the duration of $\overline{M}$. Although currently only feasible allocations are considered, any of the relaxed `AssignMission` instantiations could be used.

Once all feasible combinations have been determined, the candidate providing the largest overall reduction in airplane flying time is selected. Figure 4 summarizes the procedure.

Although `CombineMission` allows only two missions to be combined at a time, note that the combination of larger numbers of missions can be accomplished by recursively applying the algorithm to previously combined missions.

CombineMission(M, MaxLay, MaxDist, ReqRed)
  Generate Possible Combinations:
    For each Possible Combination Pair (M_1,M_2):
      Generate Mission $\overline{M} = M_1 + M_{1,2} + M_2$
  For each $\overline{M}$ in Possible Combinations:
    If `AssignMission(`$\overline{M}$`,feasible)`
      Then add $\overline{M}$
  Apply $Eval_{MaxTripRed}$ to Feasible
End

Figure 4: Mission Combination Procedure

Linking Tanker Missions

The linking of tanker missions to airlift missions is similar to mission combination. For a mission requiring air refueling, the algorithm first tries to find an existing tanker mission that is already scheduled to be in the vicinity of the requested refueling track in the time period requested. In the event that multiple tanker missions are found, the mission that minimizes the perturbation in both missions will be selected. Failing to find an existing mission, a new tanker mission will be created and linked to the requesting mission if there is available tanker capacity.

The search template for air refueling can be expressed as $<Gen_{PossRefuel}, Gen_{FeasRefuel}, Eval_{MinPert}>$. Similar to the Mission Combination, the operator $Gen_{PossRefuel}$ corresponds to the planning component. It will search for the set of currently planned refueling missions that could possibly service a certain mission M. The linkage of tankers will typically require additional adjustments of start and end times of both missions to guarantee the synchronization of the legs and resources involved in the air refueling activity. This is achieved by establishing temporal constraints between the activities representing fuel supply and reception. The time bounds of both missions involved are then updated by propagating these constraints to the entire mission itinerary.

In contrast to Mission Combination, the "scheduling" component here, $Gen_{FeasRefuel}$, is composed of two instances of `AssignMission`, to make airlift and tanker wing assignments respectively. Once the set of possible refueling missions has been identified, and the time bounds of the missions have been adjusted accordingly, $Gen_{FeasRefuel}$ will generate the set of time intervals for which both the airlift and tanker resource are available simultaneously during the entire duration of the refueling. If there are consistent wing assignments for both missions, the pair will be marked as feasible.

After all feasible pairs have been identified, the evaluation function is used to select the pair that minimizes perturbation in the missions' itineraries. If no feasible pair is found, and there is enough tanker capacity, a new refueling mission serving the airlift mission is created.[1] Figure 5 summarizes the air refueling linkage procedure.

Utilizing the Core Components

The three core procedures discussed above, `AssignMission`, `CombineMission` and `LinkTankers`, are composed in various ways to provide user-level functionality. Each can be invoked individually on a

[1] With regard to current intended use, the creation of a new mission is equivalent to reporting failure. The creation of a dummy mission is the mechanism that can be used to notify the planner that there is a mission that cannot be refueled giving the existing set of missions.

```
LinkTankers(M)
  Generate Possible Refueling Links:
    Find possible pairs (M,R)
    Adjust and propagate time bounds for pair (M,R)
  For each Possible Refueling pair (M,R):
    If AssignMission(M,feasible)
    and AssignMission(R,feasible)
      Then add pair (M, R) to Feasible.
  If Feasible is not empty
    Then apply Eval_MinPert to Feasible
    Else generate new refueling mission
End
```

Figure 5: Air Refueling Linkage

given selected mission; this is the finest granularity mode of interaction with the system. More typically, however, user action is taken relative to some selected set of input missions. The `AssignMissions` procedure (Figure ??) provides a basis for simultaneously allocating resources to some set of missions under a given set of constraint relaxation assumptions. Analagous composite procedures are similarly defined and provided for `CombineMission`, to allow overall compression of resource usage over some interval, and for `LinkTankers`, to allow simultaneous treatment of a set of unsatisfied air refueling requests.

At present, the configuration of functional capabilities for mission allocation, combination and linkage for refueling is under user-control. Typically, `AssignMissions` is used in feasible assignment mode to construct a base allocation, interleaving the use of an `AssignTankers` composite as needed to reconcile refueling requirements. `CombineMission` (or its composite counterpart `CombineMissions`) are then applied in conjunction with various relaxed forms of `AssignMission` to iteratively improve resource usage and accommodate additional, lower priority missions. One area of current work is the design of iterative improvement search procedures for automating this process.

Implementation and Status

The scheduling engine of the Barrel Allocator is implemented in Allegro Common Lisp 5.01. The user interface is in Java 1.2. The current version runs on Windows NT platforms. The core mission scheduling procedure is quite efficient – a 2 week interval of missions extracted from the current Corporate data base (approximately 1000 missions, 5000 flights) is scheduled from scratch in less than 20 seconds on a Pentium II 400MHz. Incremental planning and scheduling actions are executed in real-time.

With support from Logicon Corporation, the principal developer of AMC's Consolidated Air Mobility Planning System (CAMPS), the NT version of the Barrel Allocator has been integrated to communicate transparently with other CAMPS components and to support AMC's overall business process. Connections to the AMC Corporate Data Base and to other CAMPS tools are accomplished through a COM interface. The Barrel Allocator component is implemented as a COM server that can be called from any client in CAMPS. It also makes use of several COM servers available in CAMPS.

The Allocator loads resource availability and mission description data from the AMC Corporate Database using one such COM server server provided by CAMPS. This server makes the queries to the database and translates mission and resource information into a format the Allocator can use. The user specifies a time interval, and a certain set of air bases and aircraft types of interest. The system will then, through the COM server, query the database for nominal resource availability levels (i.e., numbers of contract, fenced and possessed aircraft) and all missions requiring the use of those resources during the specified time interval.

To support continuous daily operations, the missions retrieved from the database are marked partitioned into three possible states:

- *New missions:* missions newly created or modified by the planner and pending approval by the Barrel.
- *Approved Missions:* missions that the Barrel and the planner have reached consensus and for which resources have been allocated.
- *Revised Missions:* missions that the Barrel has allocated resources for under relaxed assumptions, but the planner has not yet concurred.

Since *approved* and *revised* missions are the missions the Barrel has previously allocated, they will be automatically assigned, during load time, to the wing and time interval specified in the request. The actual resource availability level is obtained by reducing the nominal availability by the amount of capacity reserved by already allocated missions. The system will flag a conflict if changes in nominal resource availability cause a wing to be unavailable for already assigned missions. These problematic missions will be marked as *unassignable* and will require some user guided action to be reintroduced into the current schedule. Once the barrel has finished working with a set of missions, s/he can commit the decisions back to the database. Missions that have had their flying times or assigned resources changed by the barrel, will be marked as revised, pending approval or further change by the relevant planning office. Subsequently cancelled missions – missions that the planner has decided will no longer be supported – will be removed from the schedule, raising the opportunity to reassess other, previous constraint relaxation decisions.

The most recent version of the Barrel Allocator was transferred to AMC in mid October 1999 for alpha testing. It is expected to be released as an operational component of CAMPS 2.0 by March 2000.

Conclusions

In this paper we have described Barrel Allocator, a mixed-initiative system for day-to-day management of airlift and tanker resources. The system has been designed to provide a range of mission scheduling and planning capabilities, including incremental generation of feasible resource assignments for pending missions, generation of alternative allocation options in case of resource contention, identification of opportunities for more efficient aircraft utilization though mission combination, and generation and synchronization of tanker missions to meet air refueling requirements. Each of these capabilities can be utilized with different degrees of user control over the decision-making process, ranging from user-controlled option generation to fully automated scheduling and planning processes.

The Barrel Allocator implementation derives from the architectural principles, scheduling ontology and associated class library of the Ozone scheduling framework, which consolidates the results of application building experiences in a number of similar problem domains. Although, any new problem domain brings unique requirements and constraints that make it difficult to use pre-existing solutions "out of the box", this starting point has nonetheless substantially accelerated our efforts to develop and transition the Barrel Allocator application. The core procedure for mission allocation, for example, was initially prototyped as a fairly direct instantiation of a search template previously developed in another transportation scheduling context, and allowed rapid development of initial mission combination and linkage capabilities. Later on, the architectural and configuration flexibility of the underlying framework has allowed us to efficiently refine this functionality and to quickly respond to requirement changes resulting from increased user exposure and evolving AMC business processes and policies.

domains(Becker 1998) have flexibility

Concerning the effort involved in this transition, the greatest challenge has been, and remains, to create a bridge between the current culture and business processes at AMC, and how such processes and culture can and should evolve as the Barrel Allocator becomes fully operational. To gain user acceptance it has been necessary on one hand to demonstrate the ability to support current processes. On the other hand, it has been necessary to demonstrate alternative business processes that better exploit the potential of the Barrel Allocator to improve decision-making within AMC.

As the Barrel Allocator transitions into operation at AMC, several extensions to system functionality are currently planned. One near-term extension will extend the system's resource allocation mechanism to additionally consider air crew capacity and availability constraints (currently we enforce only constraints on crew duty day and rest requirements). A second direction of future work is to expand the system's "reactive" capabilities and support shorter-term response to exceptional execution events. Much of the system's core techniques are directly applicable to this real-time, execution management process. A third direction of planned extensions concerns evolution of the interactive option generation process. One particular interest is in developing better techniques for visualizing and comparing the impact of change.

Acknowledgements

The current Barrel Allocator system represents the cumulative efforts of several individuals. Dirk Lemmermann, Gary Pelton, David Hildum, Mark Shieh, and Seppo Torma have all made substantial contributions to the implementation. Mark Burstein has been instrumental in the definition of protocols for integration with the AMC Corporate data base, and in steering the overall integration effort. Brian Gloyer, Lindy Resner and others at Logicon have provided great support in interfacing with various CAMPS system components, and in understanding user requirements. 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 contracts F30602-97-2-0227 and F30602-96-D-0058 and by the CMU Robotics Institute.

References

C. Barnhart, N.L. Boland, L.W. Clarke, E.L. Johnson, G.L. Nemhauser, and R.G. Shenoi. Flight string models for aircraft fleeting and routing. *Transportation Science*, 32(3):208–220, August 1998.

M.A. Becker. *Reconfigurable Architectures for Mixed-Initiative Planning and Scheduling.* PhD thesis, Graduate School of Industrial Administration and The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 1998.

Clarke L.W., E.L. Hane C.A., Johnson, and G.L. Nemhauser. Maintenance and crew considerations in fleet assignment. *Transportation Science*, 30:249–260, 1996.

R.A. Rushmeier and S.A. Knotogiorgis. Advances in the optimization of airline fleet assignment. *Transportation Science*, 31(2):159–169, May 1997.

S.F. Smith and M.A. Becker. An ontology for constructing scheduling systems. In *Proceedings of the AAAI Spring Symposium on Ontological Engineering*, pages 120–129, Palo Alto, CA, April 1997.

S.F. Smith, O. Lassila, and M.A. Becker. Configurable, mixed-initiative systems for planning and scheduling. In A. Tate, editor, *Advanced Planning Technology.* AAAI Press, Menlo Park, 1996.