Proceedings of the 2015 Winter Simulation Conference
L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, eds.

HIERARCHICAL, EXTENSIBLE SEARCH-BASED FRAMEWORK FOR AIRLIFT AND SEALIFT SCHEDULING USING DISCRETE EVENT SIMULATION

Talib S. Hussain
Lisa Tiberio

Information and Knowledge Technologies Unit
Raytheon BBN Technologies
10 Moulton Street
Cambridge, MA 02138, USA

Evan VanderZee

Global Security Sciences Division
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439, USA

ABSTRACT

Due to the large size of the airlift and sealift analyses performed by the United States Transportation Command (USTRANSCOM) using the Analysis of Mobility Platform (AMP), AMP has historically used a greedy heuristic algorithm focused on specific criteria in order to be able to compute solutions efficiently. We introduce a multi-year effort to enhance the extensibility and capability of AMP to support analysts in their evolving need to explore different tradeoffs – such as the impact of different business rules or evaluation criteria – using a new hierarchical, policy-based framework that provides a fundamentally search-based approach to solving the problem. The framework applies explicit selection, traversal and evaluation policies at different levels of AMP's airlift and sealift algorithms. We present an airlift constraint scheduler capability implemented using the framework, describe ongoing prototype efforts to apply the framework across other levels of AMP, and discuss future potential enhancements.

1 INTRODUCTION

The United States Transportation Command (USTRANSCOM) uses the Analysis of Mobility Platform (AMP) to support programmatic and operational analyses to determine the feasibility of moving large numbers of cargo and passenger requirements across the globe using a wide range of transportation assets. AMP (Tustin et al. 2001) is an integrated planner, scheduler, and discrete-event simulation capable of solving a wide range of problem sizes – from hundreds of cargo and passenger movement requirements over tens of days across hundreds of assets to many thousands of requirements over thousands of days across thousands of assets. These problems, regardless of scale, must be solved rapidly (typically in minutes) as part of an analyst's workday. To achieve this efficiency, AMP has historically used a greedy algorithm designed to produce reasonable solutions for a specific set of criteria. However, as methods at USTRANSCOM have evolved, there has been an increased need to study different "what if?" scenarios, explore alternative business rules, and consider different types of tradeoffs (e.g., between cost, timeliness and resource usage). To meet these changing needs, AMP has continued to evolve as well. However, AMP's greedy algorithm has intrinsic limitations on how well it can meet these evolving needs.

In 2012, we began a multi-phase effort to generalize AMP to support a broader range of business rules and analyses (Hussain et al. 2014; Sommer et al. 2014). Our approach applies a new policy-based framework that emphasizes flexibility in problem-solving methods and facilitates extensibility to new business rules through:

1. *Hierarchical local search:* A key intuition is to view the end-to-end planning and scheduling algorithm as performing several levels of local search, where each level may use a different

search method (e.g., exhaustive versus bounded versus greedy) as appropriate based on user preferences.

2. *Explicit performance objectives:* An important capability for supporting different types of analyses focused on different tradeoffs is to enable the user to explicitly specify and vary the objectives (e.g., lower cost, earlier delivery, fewer assets, shorter routes, etc., as well as the relative importance of each) used by different levels of search to identify better solutions.
3. *Generalized data structures:* To support future extensibility to different types of analyses, a general data structure is defined at each search level in the algorithm, in turn defining a rich potential search space. For instance, AMP currently defines its greedy search for what to move next in terms of specific cargo from a single requirement. We can instead generalize this to a search to move different "aggregate loads" containing different cargo types and cargo from multiple requirements. The latter easily reduces to the former, but also supports searches that can perform load splitting or consolidation, preserve the integrity of related requirements (e.g., move all requirements for a brigade together), and more.

In this paper, we describe the current approach used in AMP to plan, schedule and simulate airlift and sealift movements and discuss some of its limitations. We introduce our hierarchical, policy-based framework and present a new airlift constraint scheduler developed based on the framework. This constraint scheduler has been deployed and is in active use. We then describe ongoing prototype efforts, planned to be completed in 2016, to extend the framework to several other levels of AMP's airlift and sealift algorithms. Finally, we discuss possible methods for further enhancing search-based capabilities in AMP.

2 CURRENT AMP APPROACH

2.1 Problem Characteristics

An AMP problem comprises a set of movement requirements, termed requirement line items (RLNs), that specify cargo and/or passengers (pax) to move that must be moved using a globally distributed set of multiple types of transportation assets (e.g., ships, fleets of aircraft, trucks and railcars) via a number of ports (e.g., sea and air) that have capacity limits (e.g., berthing/parking space and service crew limits). An RLN may have multiple types of cargo with different sizes and properties, and its total amount of cargo may require multiple ships or aircraft to move. An RLN must be moved from an origin location to a destination location, subject to certain timing constraints, such as ready to load date at origin (RLD) and required delivery date at destination (RDD). Different RLNs may also have different relative importance.

The goal of AMP is to produce detailed schedules that move all the cargo/pax as required while respecting all scheduling constraints (some of which (such as RDD) may be soft constraint), minimizing lateness at destination, and making efficient use of transportation assets. The set of constraints includes timing constraints as well as restrictions on asset usage (e.g., maximum tour length for an aircraft, valid cargo types for the aircraft), port resource usage (e.g., don't exceed available parking – termed maximum on ground (MOG) – or ground crew servicing capacity), flight crew usage (e.g., don't exceed crew duty day limits by resting or swapping crews), routes (e.g., avoid no fly zones) and more

The military transportation scheduling problem solved by AMP is an instance of the NP-hard class of problems known as Pickup and Delivery with Time Windows (PDPTW) (Dumes, Desrosiers, and Soumis 1991), a variation of the Vehicle Routing Problem with Time Windows. It also has several additional complexities, such as heterogeneous fleets (Golden et al. 1984), split loads (Nowak, Ergun, and White 2008) and multiple objectives (Grandinettia et al. 2014). Such problems (typically much simpler and smaller than AMP's) have been solved by others in various ways using heuristics, optimization and/or meta-heuristics (Wassan and Nagy 2014). However, the use case, complexity and scale of AMP have historically required a fast, heuristic greedy algorithm to meet runtime needs.

2.2 Discrete Event Simulation

For a given scenario, AMP performs a discrete event simulation in which it plans RLN movements over time and then executes those movements as the simulation clock advances, refining the plans and detailed schedules as events execute. This approach enables it to model the complex interactions between all the different modes of transport (e.g., truck to air to rail), as well as all the interactions that occur within the scenario's specific network of transportation assets and resources. Scripted simulation events in the scenario may add or change RLNs, change port or fleet availability and more at various times. While AMP does simulate many modes of transportation, our effort focuses on the *strategic* leg of AMP, which moves cargo between locations that have stretches of ocean between them. AMP simulates strategic movements by performing a two phase operation on each simulation day (see Figure 1).

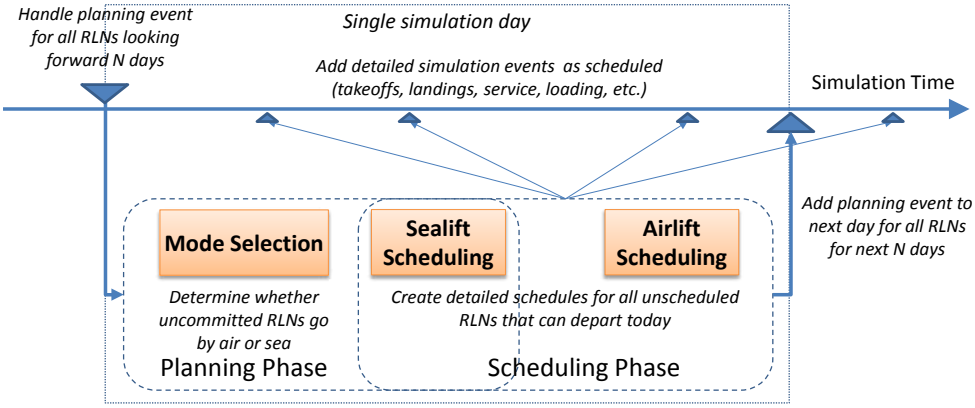

Figure 1: Illustration of strategic planning and scheduling phases on each simulation day.

2.3 Strategic Planning Phase

First, AMP performs an initial planning operation, known as mode selection, which determines whether cargo will move on aircraft or ships. For several reasons, including reducing the potential impact of the loss of an aircraft or ship, it is desirable to have groups of related cargo travel together. In AMP, this desire is represented by a constraint that all of the cargo from an RLN must travel by the same mode and between the same ports on the strategic leg. Thus, the mode selection problem can be described as the choice for each RLN of whether the RLN will travel by air or by sea. The mode selection algorithm looks out a certain number of days and considers all RLNs that have not yet been committed to a mode.

In the current AMP, the best choice of mode for an RLN is the choice that will deliver the RLN to its destination most quickly. Prior to the mode selection heuristic being run, the RLNs are put in order from highest priority to lowest priority, with RLNs that are supposed to be delivered earlier given higher priority**.** Since the goal of the heuristic is to deliver cargo as fast as possible, and aircraft move faster than ships, airlift planning is run first and plans to move every RLN that it can, starting with the highest priority RLN and working its way down to the lowest priority RLN. This is essentially a preliminary call to the airlift scheduler (see below). As airlift planning is run, it records for each RLN the estimated time that all of the RLN's cargo will be delivered. Then sealift planning begins. Starting with the highest priority RLN and working its way down to the lowest priority RLN, sealift planning looks at each RLN and computes an estimated time that it could deliver all of the RLN's cargo. This is essentially a preliminary call to the sealift scheduler (see below). If sealift planning can deliver the RLN's cargo before the estimated time that airlift planning could deliver the RLN's cargo, then a mode of sea is chosen for the RLN. At this point in the process, it remembers which ship(s) has been allocated for that RLN. When mode selection is complete for all RLNs, aircraft allocations for RLNs chosen to go by air are discarded but ship allocations for RLNs chosen to go by sea are retained.

2.4 Strategic Scheduling Phase

Once the planning has been performed, all RLNs that can be started on the current simulation day are scheduled in detail as appropriate for the planned mode. An RLN is considered committed to the current mode selection only when an actual simulation event has been generated for it. If an RLN is not ready for scheduling on the current simulation day (e.g., the cargo is not available at the port of embarkation (POE) yet), or if it is even too early to initiate supporting movements (e.g., still early to move the cargo from the origin to the POE by truck), then the RLN remains uncommitted. All uncommitted RLNs and all committed RLNs with remaining unscheduled movements are re-considered the following day (including new mode selection decisions based on the latest simulator state).

2.4.1 Airlift Scheduling

To schedule the RLNs planned to move by air on a given day, AMP's current strategic airlift scheduling capability applies a multi-step greedy algorithm, illustrated in Figure 2. For each RLN, in priority order, the scheduler tries to schedule as much of its cargo to move as possible across, potentially, multiple aircraft. Cargo is scheduled from largest to smallest, in an iterative manner, until no more cargo remains on the RLN or no new movements were successfully found. In each iteration, the algorithm checks the following options in order until the first successful mission is found. That mission is then loaded with as much cargo from the RLN as possible, and the next iteration begins.

1. The earliest matching existing mission that travels to ports are valid POE and port of debarkation (POD) for the RLN at suitable times to meet constraints (i.e., after the cargo is available to load).
2. The existing mission that can reach the POE soonest (after its last download) and can have the required RLN movement appended to its current schedule. For each potential appended mission, it tries to find a valid route from POE to POD that obeys all constraints.
3. The home-based aircraft, of the right type to carry the cargo, that is home-based closest to the POE and able to fly a new mission. For each potential new mission, a valid route is sought.

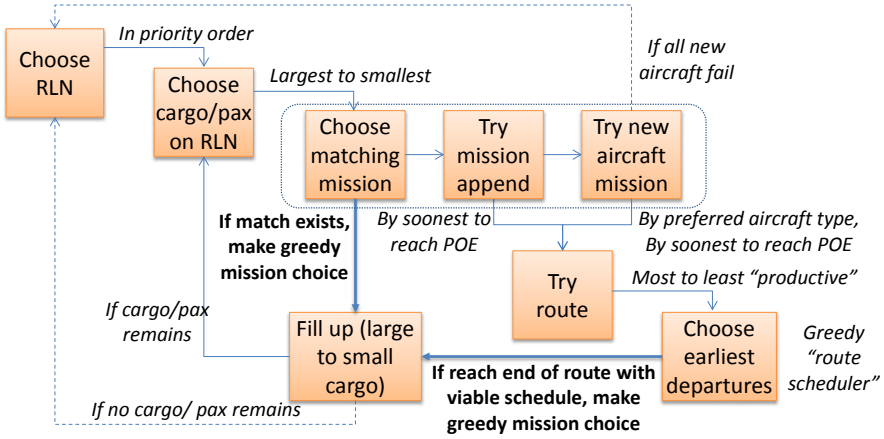

Figure 2: Illustration of key sequence of air scheduling steps and greedy choices in current AMP.

For appended and new missions, finding a valid route involves first computing possible routes from the POE to POD and sorting them according to their *productivity* (the maximum cargo load the aircraft can carry over the route divided by the total time to travel the route). For a given route, a greedy *route scheduler* tries to schedule the departure times at each successive airport on the route to be as early as possible, taking into account the availability of parking/MOG, service resources, and other constraints. It

also rests crew as late in the route as possible. In this approach, an aircraft minimizes unnecessary waiting and arrives in the earliest valid parking spot at the next port (where a valid spot is one that is long enough to meet ground time constraints at that port). If an aircraft cannot depart from any time in its slot to arrive in a valid parking slot at the next port, the route is considered a failure and the next route is tried.

2.4.2 Sealift Scheduling

To schedule the RLNs planned to move by sea beginning on a given day, AMP's current strategic sealift scheduling capability applies a heuristic that searches for suitable existing ship voyages and (in turn) new voyages, greedily loading cargo on the eligible ships it finds. First, for each RLN, in priority order, the heuristic checks whether the RLN can be scheduled on existing voyages. Then, for each remaining RLN, in priority order, the heuristic iterates over all possible POEs for the RLN and all possible PODs for the RLN. POEs are iterated over in order of increasing distance from the origin and PODs are iterated over in order of increasing distance from the destination. Eligibility of each ship is determined by whether the ship is able to carry the RLN's cargo type and is permitted to use the POD/POE. Eligible ships are iterated over in order of increasing time of availability at the POE. An estimate is made for each ship of when it would reach the POD to unload the cargo (closure date), taking into account port operating hours and berth availability. If the closure date for the ship is acceptably early, then it is added to a short list. If the short list is non-empty once all ships are examined, then ships on the short list are ordered by their estimated closure date (earliest first), and ships on the list are loaded greedily in order with as much cargo from the RLN as possible until the cargo is fully loaded or the short list is exhausted. If all of the cargo can be loaded or certain other conditions are satisfied, then the latest estimated closure date of the ships carrying cargo is compared to the date to close the RLN by airlift and to the date to close the RLN with other sealift POE/POD pairs, and the option with the earliest estimated closure date is kept. Each time new voyages are scheduled, the heuristic iterates over the remaining RLNs in priority order attempting to fill up the new voyages. Ships that are scheduled with an insufficient amount of cargo are rejected at the end of the process, and any cargo assigned to those ships is scheduled again the next day.

2.5 Limitations of the Current Approach

AMP currently uses fixed heuristics (e.g., the order in which RLNs, cargo, missions, voyages and routes are explored) and several greedy choices (e.g., pick first successful mission, pick first successful route). These can limit certain tradeoff considerations as well as introduce certain inefficiencies, such as:

- The airlift algorithm will fly an aircraft with an existing mission from a distant port in order to pick up the RLN rather than choose a new aircraft already at that port. This makes exploring different tradeoffs in timeliness versus cost difficult.
- The greedy choice of only the earliest valid arrival parking slots can result in many failed routes that would have been viable if later slots that could avoid congestion had been chosen instead.
- The greedy choice of when to crew rest can result in resting at a congested port (thereby further exacerbating the congestion), or in resting at a port between the upload and download instead of resting on a positioning leg (before or after) – which is a poor choice when carrying pax.

3 POLICY-BASED SEARCH FRAMEWORK

Our approach to enhancing AMP is to apply a framework that makes these searches and decisions as explicit and open to easy adaptation as possible. In particular, we define several classes of policies:

- *Selection policy*: Logic that defines the set of possible solutions to explore and a basic ordering of those solutions. This policy is based on a generalized data structure that defines the search space.

- *Traversal policy*: Logic that determines how the algorithm searches the space of solutions, the specific order in which solutions are considered, and the stopping criteria for the search.
- *Evaluation policy*: Logic that defines what a good quality solution is. This policy should reflect the analysts' key priorities and the tradeoffs they are examining. Explicit evaluation policies enable analysts to adapt the behavior of the search algorithms to prefer solutions that meet new priorities. For instance, on one run analysts might emphasize minimizing closure lateness, while on another run they might emphasize minimizing the number of aircraft resources used.
- *Business rule policy*: Logic that reflects a particular organizational rule that needs to be applied under certain circumstances. For example, the rules surrounding when a flight crew must rest and how much rest must be taken are well defined.
- *Heuristic policy:* Logic that reflects a particular rule of thumb to apply based on real-life experience, analysts preferences and/or developer choices.

Taken together, selection, traversal and evaluation policies fully define the search behavior of an algorithm and capture an explicit approach to balancing the efficiency of the algorithm with the quality of the solution. Business rules and heuristics can reflect small or large decisions made during a search, and reflect aspects of the algorithm that may potentially differ under different use-cases.

3.1 Policy-Based Airlift Constraint Scheduler

Our first application of the framework was to replace the greedy route scheduler with a search-based approach built upon an inductive algorithm (see Figure 3a). Rather than making commitments to specific departure times at all ports, the algorithm generalizes to the concept of a sequence of valid departure windows. Given a departure window A from port N-1, the algorithm computes the arrival window B at the next port N based on the required travel time for the aircraft. It then identifies all the possible parking slot windows over all the ramps at port N that the aircraft could possibly land in (as per *parking selection policy*) and searches over them (as per *parking traversal policy* – such as searching by ramp first, searching across ramps, or ordering all window by their starting time). For a given parking window on a given ramp, it then makes sure the window is long enough for any required ground time constraints. Based on the *crew duty business rule policy*, it determines whether crew rest or swap is required and allowed, and checks each possible case (no change, crew rest, crew swap). For each parking window that meets crew duty needs (e.g., long enough for crew rest), a check is made for available service windows (i.e., that have duration greater or equal to required service time). For a given service time (as per a *service heuristic policy* - here a greedy choice of the soonest service window is made), it computes the possible departure window containing all time remaining after applying all ground time constraints and operating hours. The algorithm assesses each departure window based on an estimate (as per *schedule evaluation policy*) of how good the full route schedule could possibly be given optimistic scheduling from N to the end of the route. A schedule evaluation policy can enforce a single criterion, or balance multiple different criteria using a weighted function – such as the amount of crew rest, closure lateness, unnecessary wait times and mission duration. Using the estimate, it picks a departure window C to explore further (as per *departure traversal policy*) and continues the induction for port N+1. Subsequent backtracking to this level of the induction may or may not occur (as per the departure traversal policy).

When this inductive algorithm reaches the final port of the route, the result is a complete set of scheduling windows where departure at any point in any of the windows is guaranteed to reach the end of the route. For each such result, the final step is to collapse it to a specific scheduling solution by selecting specific departures and distributing any unnecessary ground wait time. As shown in Figure 3b, a *wait time heuristic policy* works backwards over the route to collapse the result and choose departure times. The late decision of specific departure times offers great flexibility in wait time policies, such as preferring longer wait times at less congested ports. The quality of the completed schedule with defined departures is assessed by the schedule evaluation policy, and the departure traversal policy either accepts the best

schedule found so far or backtracks and continues the search. Different policy choices lead to different types of searches. Greedy search, bounded best-first search, exhaustive search, and more are possible.

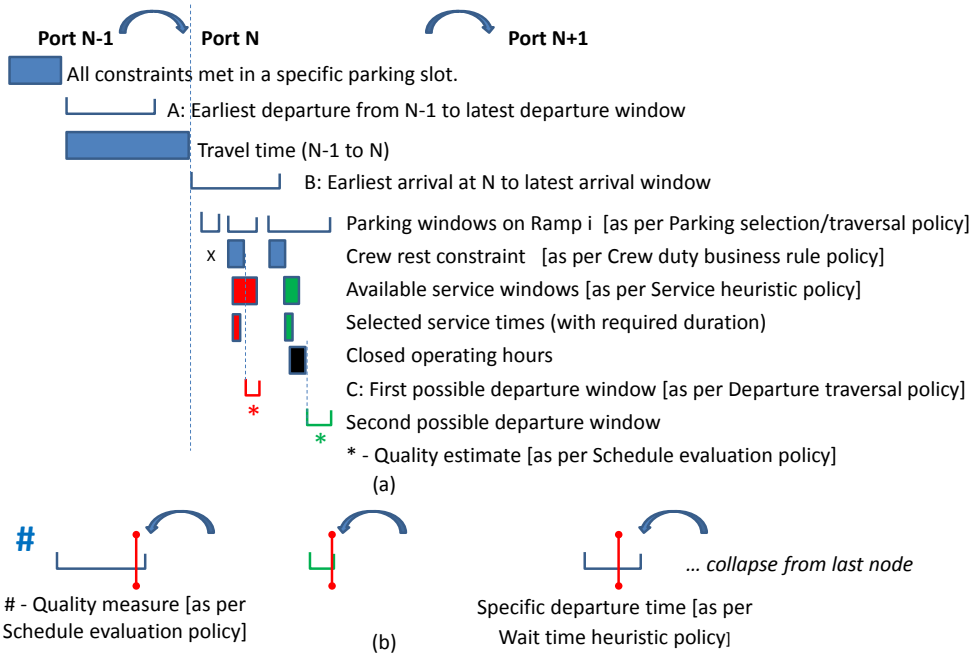

Figure 3: Policy-based airlift constraint scheduler with (a) inductive algorithm and (b) collapse method.

3.2 Results

The new policy-based constraint scheduler was deployed in 2014 at the end of our first phase of effort as an optional scheduler to use instead of the original AMP greedy core route scheduler and as part of a new AMP schedule optimization capability (Sommer et al. 2014). Full deployment of the new scheduler to replace the original one is anticipated in 2016 once it is expanded to handle additional required functionality and fully hardened. As shown in Table 1, the constraint scheduler, running using an in-depth search policy, can produce solutions that are better in solution quality than the original AMP while remaining comparable in run time for both a simple and a more complex scenario.

Table 1: Results of new airlift constraint scheduler with more in-depth search compared to AMP's original greedy scheduler.

Scenario	Scheduler	Run Time (minutes)	Closure Cargo	Closure Pax	Average Days Late	# Missions Scheduled
Simple	Original	0:10	99.99%	100%	2.38	41
	New	0:11	99.99%	100%	1.75	26
Complex	Original	3:36	99.54%	100%	3.88	3905
	New	4:05	99.53%	100%	1.88	3699

4 POLICY-BASED AIRLIFT

In our ongoing second phase of effort, we are extending the policy-based framework throughout airlift scheduling and to sealift scheduling (see Section 5). For airlift, our goal is to preserve the capabilities of

the original algorithm as a particular set of default policies while enabling a variety of alternative policies. A ground-up refactor was required in order to introduce appropriate generalizations, encapsulate state and create a hierarchy of search capabilities. We have identified and introduced more than twelve policy insertion points and four generalized data structures (see Figure 4) that, together with the completed constraint scheduler, encompass and surpass the capabilities of the original AMP.

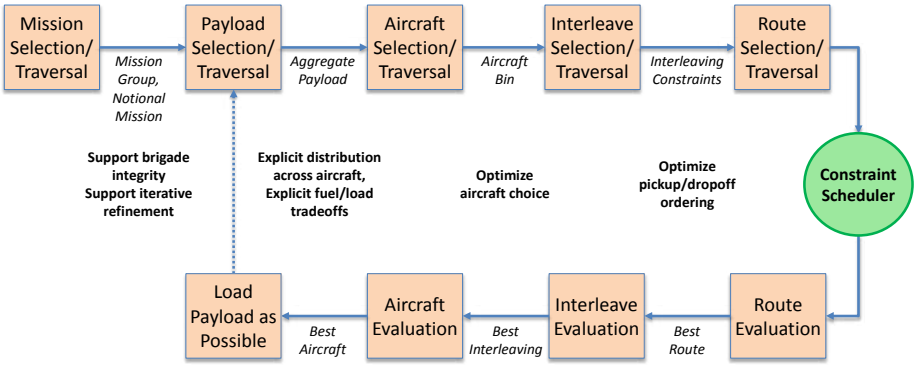

Figure 4: Five key levels of the prototype policy-based AMP airlift scheduler.

4.1 Mission Level

The *mission selection policy* determines which RLNs to consider for the current day and orders them based on an appropriate business rule. A key generalization used here is that multiple RLNs can be considered together as a *mission group*. The original AMP behavior would be equivalent to having the group always be of size 1. Multiple missions may be grouped based on similarity (e.g., share at least one possible APOE and APOD), relatedness (e.g., same brigade or operation), or other grouping logic. This supports, for example, preserving brigade integrity or ensuring full aircraft loads.

Moreover, a *notional mission* can also be provided that specifies a mission (potentially comprising multiple uploads and downloads) that can be flown by a single aircraft with additional hard or soft scheduling details (such as routing choices, schedule times and/or cargo amounts). Notional missions are also ordered with mission groups by the mission selection policy based on an appropriate business rule.

The *mission traversal policy* determines the next mission to schedule. By default, this would explore RLNs in the same order as the original AMP (i.e., furthest travel to nearest). But, new alternatives are possible, such as considering missions based on either their total amount of cargo to move, timing constraints or geographic location.

4.2 Payload Level

Given a mission to move, the *cargo selection policy* determines exactly what to try to load on a plane. A key generalization used here is that multiple cargo types and amounts (from RLNs in a mission group) can be split up into *aggregate payloads*. The partitioning up of the entire mission group into multiple payloads enables explicit balancing of load sizes against the allowable routes and the numbers and types of aircraft. In the original AMP, each aircraft for an RLN is scheduled sequentially and greedily bin-packed to capacity if possible. Due to range/payload curve limitations, the choice of how much is loaded on an aircraft impacts what routes it may fly and how many en-route stops are needed for refueling. A *cargo selection policy* can explicitly distribute all the cargo for a mission group across aircraft to enable control over the length of routes chosen and the numbers of aircraft used.

The *cargo traversal policy* determines the order in which to try to schedule aggregate payloads and how extensively to try to schedule a given mission group before moving on to the next group. This enables balancing dedicated treatment of RLNs with more distributed treatment (e.g., depth-first versus

breadth-first traversal of the mission groups). This can be important in managing resources when two groups compete for, say, a common port. Rather than stacking all of one mission group's flights first followed by the second group's flights, they can instead be interleaved one flight at a time.

4.3 Aircraft Level

The *aircraft selection policy* specifies which aircraft or set of aircraft to consider next. A key generalization used here is that of *aircraft bins*. An aircraft bin represents a set of aircraft that each has, in principle, a competitive reason to be considered for the solution. In a binned approach, all aircraft (for new and existing missions) may be ranked against multiple different criteria. (e.g., by distance from the APOE, by amount of remaining cargo carrying capacity, by how far along in their total mission duration they are). A bin can then be formed by choosing, in turn, the next best aircraft from each criterion.

The *aircraft traversal policy* can independently try out all the aircraft in the bin and evaluate the resulting scheduling solutions against an aircraft evaluation policy. The aircraft bin generalization supports parallel search and more fine-grained control by analysts over how aggressively high-level objectives (embodied in the evaluation function) are pursued. This would allow, for instance, preference for flying a new mission with a nearby aircraft over a distant aircraft with an existing mission when the former is evaluated as less costly (for a particular cost-based evaluation function). The traversal policy can explore the bins as deeply as desired based on a stopping condition (e.g., greedily to exhaustively).

4.4 Interleave Level

The *interleave selection policy* specifies a particular interleaving of uploads and downloads in order to fly a given mission with a given aircraft. A key generalization used here is that of an *interleaving constraints specification*. At a minimum, the specification indicates all of the upload and download stops, their ordering and their timing constraints. Different interleaving of uploads and downloads (e.g., up, up, down, down vs. up, down, up down) may be possible – this policy is not generating a full route, but rather providing alternative approaches to flying the mission. With this approach, different ways of aggregating cargo across different uploads can be explored, rather than the append-only approach of the original AMP. Also, when a *notional mission* is provided, the specific interleaving from that earlier solution can be preserved or improved upon depending on the policy.

The *interleave traversal policy* performs a search over the set of possible interleavings (which may range from sequential and greedy to exhaustive and parallel). Each specification informs the route selection/traversal policies what the required stops are, and the route policies can then explore a variety of alternative en-routes. Based on an interleave evaluation policy, the best interleaving is selected.

4.5 Route Level

Given an interleaving constraints specification, the *route selection policy* creates a range of possible routes by composing alternate routes between any two successive pairs of stops on a route. The *route traversal policy* explores those routes with an explicit ordering and explicit stopping criterion. This enables us to explore alternatives to the basic productivity ranking, such as accounting for current levels of congestion in the network (e.g., prefer routes that are less congested).

4.6 Hierarchical Evaluation Policies

At each level of search, different tradeoffs may be at play. Our approach seeks to align the evaluation policies used at each level to ensure top-to-bottom consistency where possible to achieve the analysts' key objectives. For instance, choices at a low level should not be in competition with choices at a higher level - both should account for related criteria when possible. For example, if the low level always skips a lower cost solution, then the higher level cannot truly optimize for cost, but if both include a weighted function balancing cost with lateness, then end-to-end optimization of cost is improved.

5 POLICY-BASED SEALIFT

As part of the ongoing second phase of the effort, we are creating a prototype that generalizes the portion of the sealift heuristics that, given an RLN that cannot fit on any existing planned voyages, searches for one or more new voyages that are capable of carrying the RLN. Figure 5 illustrates the design for how the hierarchical policy-based framework will be applied to this part of the sealift scheduling problem.

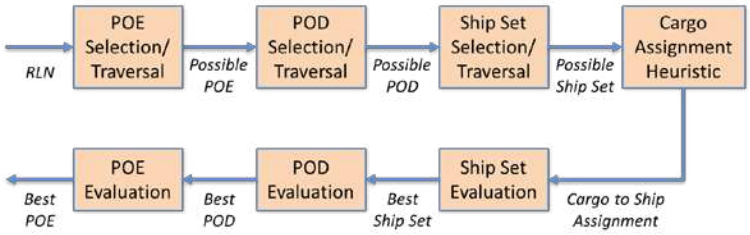

Figure 5: Illustration of policy-based sealift planning.

The top two levels – the POE and POD policies – will give the analyst control over the order in which pairs of ports are presented to the rest of the algorithm. Analysts will be able to choose to order ports by increasing expected busy time or by increasing distance from POE to POD rather than the distances from origin to POE and POD to destination. The next level – the ship set policies – generalizes the current short list of eligible ships with acceptably early closure dates, giving analysts the ability to limit the number and dissimilarity of ships passed to cargo assignment at the same time. The cargo assignment heuristic policy generalizes the current greedy assignment of cargo to ships with the earliest estimated delivery dates, giving analysts the ability to explore assigning cargo to the latest ships that are on time, assigning cargo to cargo areas of the ship in different ways, assigning cargo to ships with maximum possible utilization, and so on. Policy-based sealift, like policy-based airlift, will include hierarchical evaluation policies that give analysts the ability to align the evaluation criteria used at different levels of the algorithm, making all levels of the search consistent with the analyst's current key objectives.

6 FUTURE DIRECTIONS

Once we have completed the current effort to prototype policy-based generalizations in the airlift and sealift scheduling algorithms, there are several potential future steps.

6.1 Alternative Search Levels

In our current application of the framework, the order of the decisions made mirrors the order in the original AMP algorithm. However, this ordering is itself a heuristic, and alternatives are possible that may introduce different ways of solving the problem and/or different efficiencies. For instance, rather than scheduling one aircraft at a time and performing independent routing decisions for each aircraft, a more deliberate policy could analyze the total required number of aircraft needed and make a more global estimate of the best routes and frequency of missions to use. This could, for example, isolate different subnetworks that may be used for different RLNs and maximize throughput along those subnetworks.

6.2 Iterative Simulation

AMP currently operates in a linear, non-iterative fashion. Within a run, the simulation moves forward one day at a time. Choices made early in the run can highly constrain what happens on later simulation days, but those early choices cannot be revisited. Between runs there is only one limited form of manual iteration. Analysts can lock an air mission for a previous run, and the mission will be executed the same way in a new run, using the same aircraft type, route, scheduled times and cargo loads.

An opportunity for iteration may become possible with a policy-based framework extended through mode selection, airlift, and sealift. After an initial simulation run or portion of a run is completed (without executing scripted simulation events that are supposed to be surprise events) some missions can be retained and provided automatically as notional missions that the subsequent run may replicate or try to improve. The choice of missions to retain could be policy-based, enabling the analyst to focus on, say, iteratively lowering the cost of the overall solution by keeping low-cost missions.

6.3 Policy-Based Mode Selection

The mode selection heuristic currently used in AMP considers one RLN at a time and estimates whether the RLN can be delivered earlier by sea than by air. If sea delivers earlier, the process also assigns the RLN to specific ships, though those ships might be better utilized for other RLNs that haven't been considered yet. A policy-based approach could support a more general search that explores potential mode choices across all RLNs before assigning any RLN to any specific ship. Moreover, the planning and scheduling phases could be fully detangled, as depicted in Figure 6.

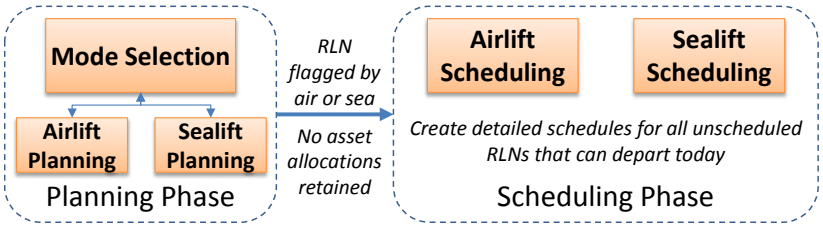

Figure 6: Illustration of policy-based mode selection as fully independent from scheduling.

A selection policy within mode selection could control the order in which groups of cargo (e.g., RLNs) are considered by the mode selection heuristic. In the current algorithm, the order is determined by the RLN priority mentioned in Section 2.3. Alternative policies such as ordering the groups of cargo from largest to smallest or ordering them by proximity in time and location could be considered.

A traversal policy within mode selection could explore different combinations of mode assignments across RLNs. The current heuristic assigns a mode of airlift to all RLNs and then considers changing that mode to sealift for one RLN at a time in priority order. An alternative traversal policy could be to explore changing *N* RLNs at a time. For example, changing two RLNs at a time, with four RLNs total, would result in comparing (air, air, air, air) to (sea, air, air, air), (air, sea, air, air) and (sea, sea, air, air) and picking the best pair of mode choices for the first two RLNs before proceeding.

An evaluation function within mode selection using different objectives could make it possible to perform a wider range of analyses with AMP. The current approach focuses almost exclusively on lateness. However, cost is another important consideration for analysts. An alternative evaluation policy for mode selection, based on Jason Judd et al. (2014), could be to convert lateness into cost with user-specified factors for how much it costs for each RLN to be delivered late and add actual cost.

7 CONCLUSIONS

We have introduced a policy-based framework that we have applied at multiple levels of the AMP algorithm in order to support an enhanced, extensible search-based approach. A new airlift constraint scheduler demonstrating the value of the framework has been completed. Work is underway to prototype the use of policies throughout most of AMP airlift and sealift scheduling, and is planned to complete in 2016. Once in place, the key benefit will be the ability to readily explore a range of alternative business rules and search methods. We envision that this will open up new use cases, as well as greatly enhance the variety and depth of analyses that may be performed by USTRANSCOM analysts.

ACKNOWLEDGMENTS

We would like to thank Ray Tomlinson, Jeffrey Tustin and Benjamin Bromberg for their contributions to the development of the policy-based airlift constraint scheduler. We would like to thank Peter Matthews and Thomas Whieldon at the USTRANSCOM Joint Distribution Process Analysis Center for their support of this work. This work was sponsored under USTRANSCOM Contracts No. HTC711-10-S002-0009 and No. HTC711-14-D-D001-0005. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

REFERENCES

Dumes, Y., J. Desrosiers, and F. Soumis. 1991. "The Pickup and Delivery Problem with Time Windows." *European Journal of Operational Research* 54:7–21.

Golden, B. L., A. A. Assad, L. Levy, and F. G. Gheysens. 1984. "The Fleet Size and Mix Vehicle Routing Problem." *Computers and Operations Research* 11:49–66.

Grandinettia, L., F. Guerrierob, F., Pezzellac, and O. Pisacanec. 2014. "The Multi-Objective Multi-Vehicle Pickup and Delivery Problem with Time Windows." *Procedia - Social and Behavioral Sciences* 111:203–212.

Hussain, T.S., J. Tustin, K. Rogers, and R. Shapiro. 2014. "Optimizing Simulation Parameters for Large-Scale Air Mobility Planning." *82nd Military Operations Research Society (MORS) Symposium*, Presentation #652.

Judd, J. D., S. Sommer, G. Grindey, E. B. VanderZee, C. N. Van Groningen, and J. Tustin. 2014. "Integrating Discrete Event Simulation and Mixed Integer Programming: An Industry Application." In *Proceedings of the 2014 Industrial and Systems Engineering Research Conference*, 3747–3756.

Nowak, M., Ö. Ergun, and C. C. White III. 2008. "Pickup and Delivery with Split Loads." *Transportation Science* 42(1):32–43.

Sommer, S., J. Judd, T. S. Hussain, and G. Grindey. 2014. "End To End Modeling (ETEM): Scheduling Optimization." *82nd Military Operations Research Society (MORS) Symposium*, Presentation #591.

Tustin, J. P., W. F., Ferguson, C. N. Van Groningen, C. J. Keyfauver, and E. R. Beeker. 2001. "Integrating MIDAS and ELIST into the AMP HLA Federation." *Fall 2001 SISO Simulation Interoperability Workshop, Logistics and Enterprise Models Forum*, Paper #01F-SIW-057.

Wassan N. A., and G. Nagy. 2014. "Vehicle Routing Problem with Deliveries and Pickups: Modelling Issues and Meta-Heuristics Solution Approaches." *Int. Journal of Transportation* 2(1):95–110.

AUTHOR BIOGRAPHIES

TALIB S. HUSSAIN is a Senior Scientist in the Information and Knowledge Technologies business unit at Raytheon BBN Technologies. He holds a PhD in Computer Science from Queen's University at Kingston. His research interests include transportation logistics optimization, evolutionary computation and simulation-based/game-based training. His email address is thussain@bbn.com.

EVAN VANDERZEE is an Assistant Software Engineer in the Global Security Sciences Division of Argonne National Laboratory. He holds a Ph.D. in mathematics with a computational science and engineering option from the University of Illinois at Urbana-Champaign. His research interests include mesh improvement and transportation planning. His e-mail address is vanderzee@anl.gov.

LISA TIBERIO is a Scientist I in the Information and Knowledge Technologies business unit at Raytheon BBN Technologies. She holds a MS in Computer Science from Rochester Institute of Technology. Her research interests include computer vision, image processing, and optimization methods for transportation logistics. Her email address is ltiberio@bbn.com.