Pre-publication Copy

C.C. Murray and M.H. Karwan, "An Extensible Modeling Framework for Dynamic Reassignment and Rerouting in Cooperative Airborne Operations," *Naval Research Logistics*, 57 (7), 634-652, 2010.

http://dx.doi.org/10.1002/nav.20427

Re-submitted to *Naval Research Logistics*; June 20, 2010

An Extensible Modeling Framework for Dynamic Reassignment and Rerouting in Cooperative Airborne Operations

Chase C. Murray
Department of Industrial & Systems Engineering, Auburn University, Auburn, AL 36849

Mark H. Karwan
Department of Industrial & Systems Engineering, University at Buffalo, Buffalo, NY 14260

Abstract: Unmanned aerial vehicles (UAVs), increasingly vital to the success of military operations, operate in a complex and dynamic environment, sometimes in concert with manned aircraft. We present an extensible modeling framework for the solution to the dynamic resource management (DRM) problem, where airborne resources must be re-assigned to time-sensitive tasks in response to changes in battlespace conditions. The DRM problem is characterized by diverse tasks with time windows, heterogeneous resources with fuel- and payload-capacity limitations, and multiple competing objectives. We propose an integer linear programming (ILP) formulation for this problem, where mathematical feasibility is guaranteed. Although motivated by airborne military operations, the proposed general modeling framework is applicable to a wide array of settings, such as disaster relief operations. Additionally, land- or water-based operations may be modeled within this framework, as well as any combination of manned and unmanned vehicles.

Keywords: UAV routing; dynamic re-tasking; MILP; multi-objective; course of action; DVRPTW

1 Introduction

Unmanned aerial vehicles (UAVs), sometimes acting in concert with manned aircraft, have become increasingly critical to the success of military operations, where they have proven to be adept at intelligence, surveillance, and reconnaissance (ISR) missions. Streaming video transmitted from a UAV to a central command unit may be used to pinpoint enemy locations, thus enabling missile strikes to be conducted with greater precision. A recent congressional report [4] recounts how Iraqi soldiers in Operation Desert Storm surrendered to a UAV because they had associated the sight of a UAV with the forthcoming arrival of missiles launched from an off-shore battleship. In addition to overseas ISR missions, UAVs may be used for domestic border patrol [26], traffic congestion monitoring [22], and cargo delivery [6]. The particular role of a UAV may be determined by the choice of onboard sensors. The most common sensors are electro-optical (EO) or infrared (IR) cameras, although sensors capable of detecting mines, chemical, and meteorological conditions are also available [4]. Additionally, UAVs are capable of firing weapons.

Although extensive effort goes into the creation of an initial mission plan (also known as an air tasking order (ATO)), the appearance of a 'pop-up' (e.g., a new threat is identified, an existing task is re-prioritized, or a new resource becomes available) necessitates the creation of an updated mission plan. The dynamic resource management (DRM) problem involves the reassignment and rerouting of airborne platforms to time-sensitive tasks in response to changes in battlespace conditions. The problem of

interest is to reassign the available resources (UAVs) to the updated set of tasks such that overall mission effectiveness is maximized and changes to the initial mission plan (ATO) are minimized. Without this secondary objective, an inconsequential improvement in mission effectiveness could lead to the undesirable outcome of wholesale changes in task assignments. A tertiary objective may be to minimize the total travel time for all resources.

In this paper we present an extensible modeling framework for the multi-objective DRM problem. This framework is designed to offer the flexibility required to handle a wide variety of task types, resources, and operating scenarios. Tasks are characterized by priority values, service duration, limits on the minimum and maximum number of resources that may perform them, and precedence relationships among tasks. Each task may have multiple time windows in which resources may be assigned. Additionally, a 'preferred' time within each window may be specified, capturing the situation where it is desirable to perform the task either early, late, or in the middle of a time window. In addition to being characterized by priority values, tasks may be classified as being either required or optional, thus allowing the consideration of mission-critical tasks and targets of opportunity.

We assume that a pre-specified number of heterogeneous fuel- and payload-capacity constrained vehicles (resources) are available. Each resource is characterized by a real-valued 'effectiveness' score that reflects the resource's relative capability of performing a given task. This feature is noticeably absent from the literature, where resources are characterized as either being able to perform a task or not, with no consideration for how well a particular resource may perform the task. Multiple bases (depots) may be specified, and each base may be characterized by the particular resources that may utilize the base.

Finally, while models that generate the initial ATO enjoy the luxury of longer planning times to address infeasibilities arising from a lack of resources or incompatible time windows, the DRM problem must be solved rapidly. As such, it is imperative that any DRM model return a solution, even if it is suboptimal. To guarantee the mathematical feasibility of our model, we introduce the concept of decision variables that act as 'infinite resources.' These fictitious resources represent the degree to which tasks cannot be performed by actual resources as requested. Analogous 'infinite bases' and 'infinite payloads' also address resources that do not have sufficient fuel capacity to arrive at a base prior to running out of fuel, and tasks that cannot receive the desired payload quantity. By including the 'infinite' concept in the model, the operations planner will receive a solution that highlights where deficiencies exist, rather than simply receiving an error message stating that no feasible solution exists.

The usefulness of the proposed model is that it serves to provide guidance for the best course of action by utilizing a centralized modeling approach. The motivation to pursue such an approach is twofold. First,

it reflects the current operational realities of the military by taking into account its highly-structured chain of command and by recognizing that many UAVs currently in service (e.g., the Predator) are not enabled for fully autonomous flight. For missions involving strike operations, rules of engagement require human involvement in the ultimate decision to fire missiles [7]. If UAVs are serving as subordinate resources for manned operations, changes to the initial mission plan may require approval by a "human in the loop" who is held accountable for the success of the mission. Secondly, given the degree of resource coordination that is afforded by the proposed model, our centralized planning approach is practical. A decentralized approach that solves separate subproblems for each UAV may result in globally suboptimal solutions or lead to violations of the coordination constraints. Infeasibilities must be reconciled by either a centralized system or through an iterative process requiring additional inter-resource communication. Conversely, in the centralized approach, only one download of the current status of each resource and one upload of the official updated mission plan are required.

Although motivated by airborne military operations, the proposed general modeling framework is applicable to a wide array of settings, such as disaster relief operations. Additionally, land- or water-based operations may be modeled within this framework, as well as any combination of manned and unmanned vehicles.

The outline of this paper is as follows. A review of relevant literature is given in section 2. In Section 3 a formal definition of the DRM problem is provided, including the notation employed by the proposed model. Section 4 details our method of classifying tasks. Section 5 contains the mathematical descriptions of nine unique task types that exploit the flexibility of our modeling framework. Payload delivery constraints are presented in section 6. Constraints governing the network structure of our model are described in section 7. After having first described the model's constraints, the multiple objective function terms are explained in section 8. A numerical example that demonstrates many model features is provided in section 9. Finally, a summary and suggestions for areas of future research are contained in section 10.

2 Literature Review

Since our work focuses on the development of a general modeling framework that is applicable to a wide variety of aircraft routing and re-routing scenarios (including disjoint time windows, heterogeneous resources, complex task types, and optional tasks), literature related to our problem cannot be limited to a particular class of problem. There are, however, several categories of problems that share commonalities with our work. The first such category is multiple-objective vehicle routing and scheduling problems.

A survey of such problems, including multi-objective variants of the traveling salesman problem (TSP), vehicle routing problem (VRP), and dynamic VRP (DVRP), is provided by Jozefowiez et al. [12]. It is worth noting that their review paper does not cite any papers that address the objective of minimizing changes to initial vehicle routes.

The second category includes classical routing problems, such as the VRP with time windows (VRPTW) and its myriad extensions. Finally, there are numerous works pertaining to UAV routing. The remainder of this section focuses on these last two categories.

2.1 Classical Routing Problems

In the VRPTW, a fleet of homogeneous resources which begin and end their routes at a single depot location, are dispatched to service customers. We may consider 'customers' to be synonymous with 'tasks.' Each customer must be served exactly once by a single vehicle, such that service at each customer must begin within a single time window, defined by the earliest and latest allowable service start times. The objective is typically to minimize travel time. The book by Toth and Vigo [24] provides an excellent overview of the vehicle routing problem and many common variants, including the VRPTW.

Extensions of the VRPTW share more features found in our model. For example, Favaretto et al. [8] extend the VRPTW to allow multiple disjoint time windows for each customer. Pellegrini et al. [17] also consider multiple time windows, as well as heterogeneous resources. In the site-dependent VRPTW (SDVRPTW) (c.f., Cordeau and Laporte [5]), each customer may be served by only a subset of the available heterogeneous resources. Unlike the aforementioned VRPTW variants, which require each customer to receive service, the so-called m-VRPTW proposed by Lau et al. [15] seeks to maximize the number of customers who receive service from one of m homogeneous resources. A secondary objective of the m-VRPTW is to minimize the total travel distance.

Another problem formulation that allows some customers to go unserved is the team orienteering problem (TOP) (c.f., Vansteenwegen et al. [25]). In the TOP, a reward is received for each customer visited, and the objective is to maximize the total reward earned, subject to a constraint on the maximum route length for each homogeneous vehicle. Goel and Gruhn [10] propose the so-called 'general VRP', which extends the TOP by considering time windows, heterogeneous vehicles, multiple depots, and an objective function that also incorporates a travel cost.

Because our problem of interest also involves real-time changes in battlespace conditions, such as the appearance of new targets, our problem is a generalization of the dynamic VRPTW (DVRPTW). Surveys of dynamic vehicle routing problems may be found in Psaraftis [18], Bianchi [3], and Flatberg

Table 1: A comparison of related UAV routing models.

Author	Dynamic	# of Objectives	Task Priority	Hetero. Resources	Time Windows	Payload Capacity	Fuel Capacity	Skip	Timing	Precedence	Max Res/Task	# of Depots
Bellingham et al. [1]	Yes	Single	No	Yes	None	No	Yes	No	No	Yes	≥ 1	Multiple
Berger et al. [2]	Yes	Multiple	No	No	Single	No	No	Yes	No	No	1	Single
Janez [11]	No	Single	No	Yes	Yes	No	Yes	No	No	No	1	Multiple
Kingston and Schumacher [13]	No	Single	No	No	Single	No	No	Yes	Yes	Yes	1	None
Kinney et al. [14]	Yes	Single	Yes	Yes	Single	No	Yes	Yes	No	No	1	Multiple
Schumacher et al. [19]	Yes	Single	No	Yes	Single	Yes[a]	Yes	No	No	Yes[b]	1	None[c]
Shetty et al. [20]	No	Single	Yes	Yes	None	Yes	Yes	Yes	No	No	≥ 1	Single
Shima et al. [21]	No	Single	No	No	None	No	Yes	No	No	Yes[b]	1	None
Tavana et al. [23]	Yes	Multiple	Yes	No	None	No	No	Yes	No	No	≥ 1	None
Weinstein and Schumacher [27]	No	Single	No	No	None	No	Yes	No	Yes	Yes	1	Multiple
Wilde et al. [28]	Yes	Multiple	Yes	Yes	Single	No	Yes[d]	Yes[e]	No	Yes	1	Single

[a] Each UAV has a single unit of expendable payload.
[b] Each target is assigned three specific tasks in succession.
[c] Each UAV begins at a unique initial location and may end at the last target location in a route.
[d] All resources have the same fuel capacity.
[e] Only tasks without time window constraints may be skipped.

et al. [9]. Li et al. [16] propose a DVRPTW model, in the context of a package delivery service, that seeks to minimize the number of initially-scheduled customers that are no longer served as a result of vehicle breakdowns. Changes to customer delivery times and changes in vehicle routes are not penalized.

While these classical routing problems share several features with our proposed model, none of them allow a customer (target) to be visited by more than one resource, none consider precedence constraints or the coordination of customers being served at the same time, and none incorporate the degree to which some resources may be more effective than others.

2.2 UAV Routing

As UAVs become more prevalent, research on UAV routing has grown in popularity. Table 1 contains a comparison of recent UAV routing papers, as they pertain to many of the key features of our proposed model. The second column of Table 1 indicates whether the paper addresses a dynamic re-routing problem or a static initial-planning problem. If a model distinguishes tasks by priorities, this is indicated in the fourth column. The column labeled 'Skip' indicates whether or not the model allows a task to be unassigned to any resource. Models that consider the coordination of multiple tasks at the same time are highlighted in the 'Timing' column. The 'Max Res/Task' column indicates the maximum number of resources that may be assigned to each task.

Although each of these papers share some similarities with our work, none demonstrate a comparable level of flexibility for solving the DRM problem. Additionally, there are features found in our proposed model that are missing from the related literature. These features include an effectiveness value for each

resource/task pair, the ability to perform tasks in non-continuous time intervals, the specification of preferred task performance times within time windows, a multi-objective formulation that incorporates the minimization of changes to an ATO, and bounds on the number of resources that may be assigned to each task. Of the aforementioned papers, only Shetty et al. [20] consider multiple payload types, and bounds on the quantity of payload that may be delivered to each target.

3 Problem Definition

In this section we establish the formal definition of the DRM problem, including characteristics of the time horizon, resources, and tasks. A description of the network underlying the DRM problem, as well as the definition of the primary decision variables employed by the model, follows.

3.1 Time Slices

The time horizon for the mission is given by the discrete (integer) set $T \subseteq \{t_0, t_0 + 1, \ldots\}$, where t_0 represents the time at which the pop-up event occurred. We refer to each element t of T as a time 'slice', where all time slices are assumed to be of equal duration and service of a task is initiated at the beginning of a time slice. Based on this partitioning of the time horizon, it is assumed that all tasks assigned to be performed in time slice t_0 are completed according to the *initial* mission plan (ATO). Therefore, time slice $t_0 + 1$ represents the first time slice in which a re-assigned plan may go into effect. It is assumed that pop-up events are recognized immediately after the beginning of a time slice (t_0).

3.2 Resources – Airborne Platforms

Let set R represent the fleet of heterogeneous resources that are available at the time that the pop-up event occurs. Resources that are no longer fit for service are not contained in R. The remaining number of time slices for which resource $r \in R$ may remain in service before exhausting its fuel supply is given by the integer value g'_r.

A set of bases (depots) is represented by B, and the set of bases that are available to resource $r \in R$ is given by $B_r \subseteq B$. Due to fuel capacity considerations, it is desirable for each resource r to end its route at a base $b \in B_r$ prior to exhausting its fuel supply (i.e., by time slice $t_0 + g'_r$). Let $T_b \subseteq T$ represent the set of allowable time slices for which resource r may visit base $b \in B_r$. Resources must loiter, without penalty, if they arrive at base b during a time slice that is not contained in set T_b.

In the event that resource r is unable to reach any available base prior to running out of fuel (e.g., if the resource were damaged or a base is no longer available), our requirement that each resource must

terminate its route at a base would be violated. To handle this situation, we define $\Delta_r^* \notin B$ to be a 'dummy' base for each resource r that does not have sufficient fuel capacity to travel directly from its current location at time t_0 to the nearest available base. We assume that the dummy base has no physical location, and that the resource can travel to this dummy base from anywhere in exactly one time slice. We let $T_{\Delta_r^*} = t_0 + g'_r + 1$ represent the time slice in which resource r may be assigned to this dummy base, thus allowing the resource to perform tasks until it runs out of fuel at time $t_0 + g'_r$ and then 'travel' to the dummy base in the next time slice.

3.3 Tasks

The set of tasks that have yet to be completed by the time of the pop-up event are given by M. Each task $j \in M$ has an associated non-negative priority weight, p_j, such that tasks with larger priority weights offer greater benefit to the overall mission plan. To capture the fact that each resource may have unique capabilities, we let $e_{r,j}$ denote the effectiveness of resource $r \in R$ performing task $j \in M$. Each $e_{r,j}$ is a real number, such that larger values represent higher effectiveness. Let $M_r \subseteq M$ denote the set of tasks that can be performed by resource r, such that task $j \in M_r$ if $e_{r,j} > 0$. Similarly, we define $R_j \subseteq R$ to be the set of resources capable of performing task $j \in M$, such that resource $r \in R_j$ if $e_{r,j} > 0$.

The set of allowable time slices in which task $j \in M$ may begin service is given by $T_j \subseteq T$. We do not require T_j to be a set of consecutive time slices. However, it is assumed that each resource may perform at most one task in any given time slice $t \in T$. As with bases, if a resource arrives at task j during a time slice that is not contained in set T_j, the resource must loiter until the next allowable time slice.

3.4 Network Structure

To facilitate the representation of our mathematical model, we make use of the following notation to describe the underlying structure of the network. We should note that the terms 'node' and 'task' are unique; a task is a particular type of node, where nodes may also represent base locations, initial resource locations, and dummy bases.

First, let Δ_r^0 be the node representing the current location (at time t_0) of resource $r \in R$. Current locations are considered to be 'source' nodes, such that resource r cannot travel *to* node Δ_r^0. Note that Δ_r^0 is neither a task nor a base node (i.e., $\Delta_r^0 \notin \{M \cup B\}$). Conversely, we consider each actual base $b \in B$ and each dummy base Δ_r^* to be 'sink' nodes, such that a resource cannot return to service after arriving at a base.

Let $\Delta_r^+ \subseteq \{M_r \cup B_r \cup \Delta_r^*\}$ represent the set of all nodes to which resource $r \in R$ may travel. Note that

$\Delta_r^0 \notin \Delta_r^+$ since resources are not allowed to travel *to* their current (source) location. Let $\Delta_{r,j}^- \subseteq \{\Delta^0 \cup M_r\}$ represent the set of all nodes from which resource $r \in R$ may travel to node $j \in \{M \cup B \cup \Delta_r^*\}$. Note that bases are not included in $\Delta_{r,j}^-$ since, by assumption, resources may not depart from a base (sink) node.

Let $f_{r,i,j}$ represent the *minimum* number of time slices that are required for resource r to travel from node i to node j. If a particular arc (i,j) in the network is impassible for resource r, then $f_{r,i,j}$ should be set to a value of ∞ and node i should be excluded from $\Delta_{r,j}^-$. The effects of known obstacles (such as mountains), inclement weather, enemy locations, or no-fly zones, may be captured by appropriately augmenting the $f_{r,i,j}$ values to reflect extra travel time between nodes. Because our model relies on resource-specific travel time between nodes, rather than distance, the locations of tasks and bases may be expressed in any coordinate system, such as Cartesian (x,y) or latitude/longitude.

Finally, $f_{r,i,\Delta_r^*} = 1$ for all $r \in R$ and $i \in \Delta_{r,\Delta_r^*}^-$. If the resource has sufficient fuel to travel directly from its starting location to the nearest base, then the resource is not allowed to get stranded at a dummy base (i.e., $\Delta_r^* \notin \Delta_r^+$ unless $\min_{b \in B_r}(f_{r,\Delta_r^0,b}) > g_r'$). Since it will be determined *a priori* whether or not a dummy base will be required for a particular resource, an operator may be warned that the resource will be stranded. The operator may then define an actual base $b \in B$ at an actual physical location for which to assign the resource to terminate its route.

3.5 Primary Decision Variables, $x_{r,i,j}^t$

Our model makes use of binary decision variables $x_{r,i,j}^t$ to determine whether or not resource $r \in R$ is assigned to travel to node $j \in \Delta_r^+$ from node $i \in \Delta_{r,j}^-$, performing the action at node j at time $t \in T_j$. That is,

$$x_{r,i,j}^t \in \{0,1\} \qquad \forall\, r \in R, j \in \Delta_r^+, i \in \Delta_{r,j}^-, t \in T_j. \tag{1}$$

Additional integer decision variables required by the model are addressed as needed.

Because our problem of interest is to solve a re-assignment problem, we assume that an initial mission plan is provided. This initial assignment of resources to tasks is represented by the binary parameter $a_{r,i,j}^t$, where $a_{r,i,j}^t = 1$ if resource $r \in R$ was initially assigned to perform task $j \in \Delta_r^+$ during time slice $t \in T_j$, such that task $i \in \Delta_{r,j}^-$ was the immediate predecessor. Otherwise, let $a_{r,i,j}^t = 0$.

4 Task Characterization

Previously we have mentioned that a mission is composed of a set of tasks, M. In this section we discuss a variety of task characteristics. In the next section we will apply combinations of these characteristics to define nine unique task types.

4.1 Required vs. Optional

First, we may characterize each task as being either *required* (mission-critical) or *optional* (non-mission-critical). A penalty is assessed for failing to perform required tasks. Optional tasks may offer some benefit to the overall mission, but there is no penalty for failing to perform these tasks. Let $n_j^{\min}$ ($n_j^{\max}$) denote the minimum (maximum) number of resources that may be assigned to perform task j, such that $n_j^{\min} \leq n_j^{\max}$. For all *required* tasks, $n_j^{\min} \geq 1$.

Because we do not know in advance whether or not the available resources are sufficient to perform all mission-critical tasks within their allowable time windows, an 'infinite' resource is included in the model. This fictitious resource is able to travel at infinite velocity, has infinite capacity, and can perform multiple tasks at the same time. The inclusion of this resource ensures that constraints requiring assignments to mission-critical tasks will always be mathematically feasible. Conversely, since optional tasks are not required, they do not need the use of an infinite resource to maintain mathematical feasibility.

4.2 Non-preemptive vs. Preemptive

Let d_j represent the required duration of task $j \in M$, expressed as an integer number of time slices. We define *non-preemptive* tasks to be those in which a resource, once assigned to the task, must continue to perform the task uninterrupted for d_j consecutive time slices. Let the set $M^{\text{non}} \subseteq M$ represent the set of all non-preemptive tasks. For non-preemptive tasks, $d_j \geq 0$. If $d_j = 0$, the task is assumed to be performed instantaneously (e.g., dropping a bomb on a target or taking a snapshot). Because we define non-preemptive tasks to be performed in uninterruptable consecutive time slices, if $x_{r,i,j}^t = 1$ for $j \in M^{\text{non}}$, then resource r begins performing task j at time slice $t \in T_j$, continuing for d_j consecutive time slices. During time slices t through $t + d_j$ resource r is considered to be busy and may not be assigned to another task. Note that T_j represents the set of allowable *starting* time slices for task $j \in M^{\text{non}}$. Thus, d_j may be greater than $|T_j|$ for non-preemptive tasks.

Because a resource assigned to non-preemptive task j must continue to perform the task for d_j consecutive time slices, the calculation of $f_{r,j,k}$ must equal the actual travel time (in number of slices) from task j to node k plus the duration of task j, d_j. A resource may not re-visit the same non-preemptive

task. For this reason, $x^t_{r,j,j}$ is not defined for $j \in M^{\text{non}}$. Similarly, $j \notin \Delta^-_{r,j}$ for all $j \in M^{\text{non}}$ and $r \in R_j$. Therefore, $f_{r,j,j}$ is undefined for all $j \in M^{\text{non}}$.

We define *preemptive* tasks to be those in which a resource is assigned to the task for one entire time slice at a time. Let $M^{\text{pre}} \subseteq M$ be the set of all preemptive tasks. Note that preemption cannot occur *within* a time slice. Preemptive tasks are assigned to resources on a slice-by-slice basis. Unlike non-preemptive tasks, if a resource is assigned to perform a preemptive task $j \in M^{\text{pre}}$ for a duration of d_j time slices, then d_j binary decision variables ($x^t_{r,i,j}$) must take a value of one.

If $x^t_{r,i,j} = 1$, and $j \in M^{\text{pre}}$, resource r is assigned to travel from node $i \in \Delta^-_{r,j}$ to task j, beginning service at task j at the beginning of time slice t, and continuing to perform service at task j until the start of time slice $t+1$. As a result, the following must be true of *preemptive* tasks. First, $x^t_{r,j,j}$ must be defined for all $j \in M^{\text{pre}}$ and j must be an element of $\Delta^-_{r,j}$ for all $r \in R_j$. This allows a resource to re-visit a particular preemptive task multiple times. Next, T_j represents the set of time slices in which task $j \in M^{\text{pre}}$ may be *performed*. Also, $d_j \geq 1$ because resources are assigned to preemptive tasks for an entire time slice. Because all d_j assigned time slices must be selected from set T_j, $d_j \leq |T_j|$, where $|T_j|$ represents the cardinality of set T_j. Finally, since preemptive tasks are assigned one time slice at a time, $f_{r,j,k} = 1+$ (travel time from task j to node k) and $f_{r,j,j} = 1$ for all $j \in M^{\text{pre}}$.

Preemptive tasks are attractive because of their flexibility (e.g., allowing a resource to leave a task to perform another task and then resume), and preemptive tasks cannot be easily modeled by the multicommodity network flow model. Non-preemptive tasks, while less flexible, require fewer decision variables (resources are not assigned to tasks slice-by-slice).

Before proceeding to the mathematical formulations of some task types, we define one more parameter that will be of use later. Let d'_j denote the maximum number of times a given resource may be assigned to node j. More formally,

$$d'_j \equiv \begin{cases} 1 & \forall\ j \in \{M^{\text{non}} \cup B\}, \\ d_j & \forall\ j \in \{M^{\text{pre}}\}. \end{cases} \tag{2}$$

This parameter is required due to the differences in the functionality of preemptive and non-preemptive tasks. Note that bases are treated as non-preemptive tasks because a resource may not revisit a base.

Table 2: Notation – Tasks

RNA	Required, non-preemptive tasks that may be performed in any allowable time slice.
ONA	Optional, non-preemptive tasks that may be performed in any allowable time slice.
RNS	Required, non-preemptive tasks that must be performed by multiple resources simultaneously.
$RNM(s)$	Set of required, non-preemptive tasks in which multiple tasks must be performed simultaneously.
RNP	Required, non-preemptive tasks with precedence requirements.
RPC	Required, preemptive tasks that must be performed in consecutive time slices.
RPD	Required, preemptive tasks that may be performed in disjoint time slices.
OPD	Optional, preemptive tasks that may be performed in disjoint time slices.
OPP	Optional, preemptive tasks that may be peformed for a partial duration.

5 Task Types

In this section we formally define nine unique task types, as listed in Table 2. To differentiate among these task types, a three-letter naming convention is adopted. The first letter, either 'R' or 'O', indicates whether the task is *required* or *optional*. The second letter, 'N' or 'P', classifies the task type as being either *non-preemptive* or *preemptive*. Finally, the third letter represents some unique characteristic about that particular task type. For example, we define four required non-preemptive, one optional non-preemptive, two required preemptive, and two optional preemptive task types.

5.1 RNA Tasks

We define $RNA \subseteq M^{\text{non}}$ to be the set of required (mission-critical) non-preemptive tasks such that each resource assigned to task $j \in RNA$ may *begin* service in any allowable time slice $t \in T_j$, with a service duration of $d_j \geq 0$ consecutive time slices. If $n_j^{\min} \geq 2$, then multiple resources are required to perform task j. However, for RNA tasks, these multiple resources do *not* need to be assigned to the task at the same starting time. The 'infinite resource' available for task $j \in RNA$ is given by the integer decision variable y_j. A non-zero value of y_j indicates the gap between the minimum required number of resources and the actual number of resources assigned to task $j \in RNA$.

The mathematical representation of RNA tasks is given by the following constraints.

$$n_j^{\min} \leq \sum_{r \in R_j} \sum_{t \in T_j} \sum_{i \in \Delta_{r,j}^-} x_{r,i,j}^t + y_j \leq n_j^{\max} \qquad \forall\, j \in RNA, \tag{3}$$

$$\sum_{t \in T_j} \sum_{i \in \Delta_{r,j}^-} x_{r,i,j}^t \leq 1 \qquad \forall\, j \in \{RNA : n_j^{\max} \geq 2\}, r \in R_j, \tag{4}$$

$$y_j \in \{0, 1, \ldots, n_j^{\min}\} \qquad \forall\, j \in RNA. \tag{5}$$

Constraints (3) state that each task $j \in RNA$ must be performed by at least $n_j^{\min}$, and by no more than

$n_j^{\max}$, resources. Constraints (4) ensure that constraints (3) cannot be satisfied by simply assigning the same resource to task j at multiple times; these constraints are only necessary if it is actually allowable to assign multiple resources to the task (i.e., if $n_j^{\max} \geq 2$).

5.2 ONA Tasks

A mission may contain some non-mission-critical non-preemptive tasks that may be beneficial in increasing overall mission effectiveness. Let $ONA \subseteq M^{\text{non}}$ represent the set of optional non-preemptive tasks that may begin service at any allowable time slice. There is no penalty associated with the failure to perform ONA tasks. Constraints (3) and (4) may be modified as noted below.

Like RNA tasks, the mission-critical counterpart of ONA tasks, it is acceptable for multiple resources to perform task $j \in ONA$ simultaneously. Similarly, it is not acceptable for the same resource to begin service at task $j \in ONA$ multiple times. Because ONA tasks are non-mission-critical, it is implied that $n_j^{\min} = 0$ for all $j \in ONA$ in constraints (4). For this reason, ONA tasks do not require the use of an 'infinite resource' to guarantee mathematical feasibility; thus y_j in (3) would not be needed.

5.3 RNS Tasks

Some missions may require a task to be performed by multiple resources simultaneously, as in the case of an air strike mission in which multiple aircraft must attack a target at the same time. We define $RNS \subseteq M^{\text{non}}$ to accommodate such a requirement. For each task $j \in RNS$, between $n_j^{\min}$ and $n_j^{\max}$ resources must begin task j during the same time slice $t \in T_j$ and perform the task for a duration of $d_j \geq 0$ consecutive time slices.

To maintain mathematical feasibility, RNS tasks make use of the integer decision variable z_j^t to represent the 'infinite resource'. Thus, a non-zero value of z_j^t indicates the gap between the number of actual resources that are assigned to begin task j at time slice t and the minimum number of resources that are required to perform task j ($n_j^{\min}$). Additionally, a binary decision variable h_j^t is used to indicate whether or not task $j \in RNS$ begins service at time slice $t \in T_j$. Mathematically, RNS tasks are governed by the following constraints:

$$h_j^t n_j^{\min} \leq \sum_{r \in R_j} \sum_{i \in \Delta_{r,j}^-} x_{r,i,j}^t + z_j^t \leq h_j^t n_j^{\max} \qquad \forall\ j \in RNS, t \in T_j, \tag{6}$$

$$\sum_{t \in T_j} h_j^t = 1 \qquad \forall\ j \in RNS, \tag{7}$$

$$z_j^t \in \{0, 1, \ldots, n_j^{\min}\}, h_j^t \in \{0, 1\} \qquad \forall\ j \in RNS, t \in T_j. \tag{8}$$

Constraints (6) ensure that the appropriate number of resources are assigned to task j during the selected time slice. Constraints (7) state that task j must be assigned to resources in exactly one time slice. Finally, constraints (8) describe the integer requirements for the decision variables.

5.4 RNM Tasks

Whereas RNS tasks require the coordination of multiple resources performing the *same non-preemptive task* at the same time, RNM tasks require the coordination of *multiple non-preemptive tasks* being performed at the same time. The RNM task type may be useful in modeling a strike mission in which multiple targets in geographically dispersed regions must be attacked simultaneously.

Let $RNM(s)$ represent the s^{th} set of RNM tasks, such that $s \in S$ is an index value, S is the set of all indices, and $RNM(s) \subseteq M^{\text{non}}$ for all $s \in S$. By indexing the RNM tasks, we allow for the fact that there may be multiple sets of the RNM task type. Consistent with our previous assertion that each task must be assigned to exactly one task type, a task may appear in no more than one $RNM(s)$ set.

Because all tasks in a given set $RNM(s)$ must begin service at the same time, it is assumed that each task $j \in RNM(s)$ has the same allowable starting time window, T_j. Thus, each task's set of allowable time slices must be defined such that $T_j = \bigcap_{k \in RNM(s)} T_k$ for all $s \in S, j \in RNM(s)$. Failure to adhere to this condition could lead to unexpected solutions.

Although each task in $RNM(s)$ must start at the same time, the *duration* of each task in the s^{th} set may be unique (i.e., $d_j \neq d_k$ for some $j, k \in RNM(s)$). Additionally, it is not a requirement that tasks $j \in RNM(s)$ and $k \in RNM(s)$ have equal minimum and/or maximum number of resources. However, $n_j^{\min} \geq 1$ for all $j \in RNM(s)$ due to the fact that RNM tasks are required (mission-critical). This flexibility is demonstrated in the example of Figure 1.

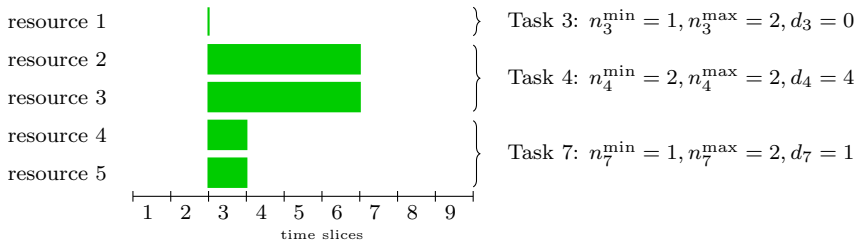

Figure 1: Resource assignments for tasks in the s^{th} set of RNM tasks, where $T_j = ([2, 4])$ for $j = 3, 4, 7$.

The 'infinite resource' for RNM tasks is given by the integer decision variable z_j^t, where a non-zero value of z_j^t represents the number of 'infinite resources' that are required for task $j \in \cup_{s \in S} RNM(s)$ at time slice $t \in T_j$. A binary decision variable, b_s^t, ensures that all tasks in the s^{th} set of RNM tasks begin

service at the same time slice, t. The constraints associated with RNM tasks are as follows:

$$b_s^t n_j^{\min} \leq \sum_{r \in R_j} \sum_{i \in \Delta_{r,j}^-} x_{r,i,j}^t + z_j^t \leq b_s^t n_j^{\max} \qquad \forall\ s \in S, j \in RNM(s), t \in \bigcap_{k \in RNM(s)} T_k, \tag{9}$$

$$\sum_{t \in \bigcap_{k \in RNM(s)} T_k} b_s^t = 1 \qquad \forall\ s \in S, \tag{10}$$

$$z_j^t \in \{0, 1, \ldots, n_j^{\min}\} \qquad \forall\ s \in S, j \in RNM(s), t \in \bigcap_{k \in RNM(s)} T_k, \tag{11}$$

$$b_s^t \in \{0, 1\} \qquad \forall\ s \in S, t \in \bigcap_{k \in RNM(s)} T_k. \tag{12}$$

Constraints (9) state that each task $j \in RNM(s)$ must be performed by the appropriate number of resources during the selected time slice. In constraints (10), exactly one time slice must be selected from the set of allowable time slices that are common to all tasks in $RNM(s)$.

5.5 RNP Tasks

UAVs are frequently employed to perform battle damage assessment (BDA), where a UAV surveils a location that has previously been the subject of a kinetic event, such as a missile strike. The proposed RNP task type, where $RNP \subseteq M^{\text{non}}$, is defined to be the set of all required non-preemptive tasks with precedence relationships. Let $RNP'(j)$ represent the set of tasks that must be completed at least lag_j time slices prior to the beginning of service of task $j \in RNP$. If none of the tasks in $RNP'(j)$ are performed, then task $j \in RNP$ should not be performed. However, if *at least one* of the tasks in $RNP'(j)$ is performed, then task j becomes a mission-critical task (a penalty is paid for not assigning the proper number of resources to perform task j). For this reason, we say that RNP tasks are 'conditionally' required. In the above scenario, task $j \in RNP$ would represent the BDA activity, while task $k \in RNP'(j)$ would represent the predecessor task of firing a missile at the enemy target. Note that $RNP'(j)$ is not a task type. As such, task k must be attributed to one of the existing M^{non} or M^{pre} task types.

Let binary decision variable $x_j^{\text{RNP}'}$ indicate whether or not at least one task from the set $RNP'(j)$ is assigned. The value of $x_j^{\text{RNP}'}$ may be set according to the following constraints:

$$x_j^{\text{RNP}'} \leq \sum_{k \in RNP'(j)} \sum_{r \in R_k} \sum_{i \in \Delta_{r,k}^-} \sum_{t \in T_k} x_{r,i,k}^t \leq \left(\sum_{k \in RNP'(j)} n_k^{\max} d_k' \right) x_j^{\text{RNP}'} \qquad \forall\ j \in RNP, \tag{13}$$

$$x_j^{\text{RNP}'} \in \{0, 1\} \qquad \forall\ j \in RNP. \tag{14}$$

In (13), $x_j^{\mathrm{RNP}'} = 1$ if at least one task from the set $RNP'(j)$ is assigned, making task $j \in RNP$ a required task. Otherwise, if $x_j^{\mathrm{RNP}'} = 0$, then none of the tasks in $RNP'(j)$ were performed, and task $j \in RNP$ should not be performed (i.e., there is nothing to assess). This relationship is captured in constraints (15) below, where y_j is the 'infinite resource' that may be used, with a penalty, to guarantee feasibility.

$$n_j^{\min} x_j^{\mathrm{RNP}'} \leq \sum_{r \in R_j} \sum_{i \in \Delta_{r,j}^-} \sum_{t \in T_j} x_{r,i,j}^t + y_j \leq n_j^{\max} x_j^{\mathrm{RNP}'} \qquad \forall\, j \in RNP, \tag{15}$$

$$\sum_{t \in T_j} \sum_{i \in \Delta_{r,j}^-} x_{r,i,j}^t \leq 1 \qquad \forall\, j \in \{RNP : n_j^{\max} \geq 2\}, r \in R_j, \tag{16}$$

$$y_j \in \{0, 1, \ldots, n_j^{\min}\} \qquad \forall\, j \in RNP. \tag{17}$$

Constraints (16) ensure that constraints (15) are not satisfied by simply assigning the same resource to task $j \in RNP$ in multiple time slices.

Thus far, we have not addressed the need for the RNP task to be performed *after* the completion of the predecessor task(s). Let the binary decision variable $x_{t,j}^{\mathrm{RNP}}$ take a value of one if task $j \in RNP$ begins service during time slice $t \in T_j$. Recall that RNP tasks are non-preemptive, so it is straightforward to ascertain the beginning of service. Constraints (18), below, are used to set $x_{t,j}^{\mathrm{RNP}}$ to its proper value:

$$x_{t,j}^{\mathrm{RNP}} \leq \sum_{r \in R_j} \sum_{i \in \Delta_{r,j}^-} x_{r,i,j}^t \leq n_j^{\max} x_{t,j}^{\mathrm{RNP}} \qquad \forall\, j \in RNP, t \in T_j, \tag{18}$$

$$x_{t,j}^{\mathrm{RNP}} \in \{0, 1\} \qquad \forall\, j \in RNP, t \in T_j. \tag{19}$$

To ensure that at least lag_j time slices elapse before beginning task $j \in RNP$, the following constraints are required. These depend on the type of predecessor task, $k \in RNP'(j)$:

$$\left(1 - x_{t,j}^{\mathrm{RNP}}\right) \geq \frac{1}{n_k^{\max}} \sum_{r \in R_k} \sum_{i \in \Delta_{r,k}^-} \sum_{\substack{t_k \in T_k: \\ t_k > t - lag_j - d_k}} x_{r,i,k}^{t_k} \qquad \forall\, j \in RNP, k \in \{RNP'(j) \cap M^{\mathrm{non}}\}, t \in T_j, \tag{20}$$

$$\left(1 - x_{t,j}^{\mathrm{RNP}}\right) \geq \frac{1}{n_k^{\max} d_k} \sum_{r \in R_k} \sum_{i \in \Delta_{r,k}^-} \sum_{\substack{t_k \in T_k: \\ t_k > t - lag_j - 1}} x_{r,i,k}^{t_k} \qquad \forall\, j \in RNP, k \in \{RNP'(j) \cap M^{\mathrm{pre}}\}, t \in T_j. \tag{21}$$

Constraints (20) state that if RNP task j is performed at time slice t, then non-preemptive task $k \in RNP'(j)$ cannot be started after time slice $t - lag_j - d_k$. In a similar fashion, constraints (21) ensure that preemptive task $k \in RNP'(j)$ cannot be performed after time slice $t - lag_j - 1$ if RNP task j is performed at time slice t.

5.6 *RPC* Tasks

We define $RPC \subseteq M^{\text{pre}}$ to be the set of required preemptive tasks that must be assigned to resources in consecutive time slices. More formally, each task $j \in RPC$ must be performed in $d_j > 0$ consecutive time slices contained in the set T_j.

Although both RNA and RPC tasks must be performed in consecutive time slices, RPC tasks are preemptive and thus offer greater flexibility because resources are assigned to RPC tasks on a per-time-slice basis. This flexibility is enabled because the binary decision variable $x^t_{r,j,j}$ is defined for all $j \in RPC$, $t \in T_j$, and $r \in R_j$. As a result, the particular resources that are assigned to each RPC task may vary over time (unlike *non-preemptive* tasks where resources are committed for the entire task duration). Figure 2 provides an example of an RPC task that must be performed in $d_j = 6$ consecutive time slices.

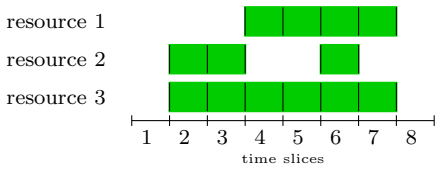

Figure 2: Resource assignments for task $j \in RPC$, where $n^{\min}_j = 2$, $n^{\max}_j = 3$, $T_j = [1, 8]$, and $d_j = 6$.

The mathematical representation of RPC tasks is given by:

$$n^{\min}_j h^t_j \le \sum_{r \in R_j} \sum_{i \in \Delta^-_{r,j}} x^t_{r,i,j} + z^t_j \le n^{\max}_j h^t_j \qquad \forall\ j \in RPC, t \in T_j, \tag{22}$$

$$\sum_{t \in T_j} h^t_j = d_j \qquad \forall\ j \in RPC, \tag{23}$$

$$\sum_{t' \in \{T_j : t' \notin [t - d_j + 1, t + d_j - 1]\}} h^{t'}_j \le (1 - h^t_j) d_j \qquad \forall\ j \in RPC, t \in T_j, \tag{24}$$

$$h^t_j \in \{0, 1\}, z^t_j \in \{0, 1, \ldots, n^{\min}_j\} \qquad \forall\ j \in RPC, t \in T_j. \tag{25}$$

Constraints (22) ensure that the appropriate number of resources are assigned to task $j \in RPC$ during each selected time slice, although different resources may be assigned to the task in adjacent time slices. In the solution, any non-zero value of z^t_j may be interpreted to represent the gap between the minimum number of resources required and the actual number of resources that were assigned to task j during time slice t. Constraints (23) require that exactly d_j time slices are selected. Finally, constraints (24) ensure that the selected time slices are consecutive. Note that constraints (24) assume that set T_j contains consecutive time slices, a reasonable assumption given the definition of the RPC task type.

5.7 *RPD* Tasks

Some tasks may not require assignment in consecutive time slices. For this reason, we define $RPD \subseteq M^{\text{pre}}$ to be the set of required preemptive tasks that may be performed in disjoint time slices. *RPD* tasks are the most flexible of all required tasks. An example demonstrating a resource assignment profile for an *RPD* task is provided in Figure 3.

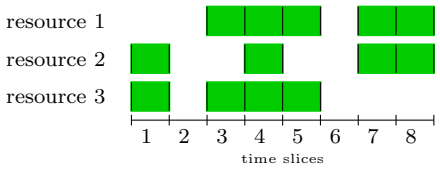

Figure 3: Resource assignments for task $j \in RPD$, where $n_j^{\min} = 2$, $n_j^{\max} = 3$, $T_j = [1, 8]$, and $d_j = 6$.

The mathematical representation of *RPD* tasks follows the same form as the *RPC* tasks, where constraints (24) are excluded. The interpretation of the 'infinite resource', z_j^t is also the same as defined for the *RPC* tasks.

$$n_j^{\min} h_j^t \leq \sum_{r \in R_j} \sum_{i \in \Delta_{r,j}^-} x_{r,i,j}^t + z_j^t \leq n_j^{\max} h_j^t \qquad \forall\, j \in RPD, t \in T_j, \tag{26}$$

$$\sum_{t \in T_j} h_j^t = d_j \qquad \forall\, j \in RPD, \tag{27}$$

$$h_j^t \in \{0, 1\}, z_j^t \in \{0, 1, \ldots, n_j^{\min}\} \qquad \forall\, j \in RPD, t \in T_j. \tag{28}$$

5.8 *OPD* Tasks

We now define $OPD \subseteq M^{\text{pre}}$ to be the non-mission-critical counterparts of *RPD* tasks. If these optional tasks are performed, they must meet the same requirements of the *RPD* tasks. For example, if performed, task $j \in OPD$ must be assigned to at least $n_j^{\min}$, and no more than $n_j^{\max}$, resources in $d_j > 0$ time slices from the set of allowable time slices, T_j. As with *RPD* tasks, the actual resources that are assigned to any task $j \in OPD$ may vary over time. However, unlike the other optional task types described in this paper, $n_j^{\min} \geq 1$ for all *OPD* tasks. This requirement results from the structure of constraints (29), below. Since these tasks are optional, there is no need for an infinite resource. Additionally, there is no penalty associated with the failure to perform *OPD* tasks. The constraints describing *OPD* tasks are as

follows:

$$n_j^{\min} h_j^t \leq \sum_{r \in R_j} \sum_{i \in \Delta_{r,j}^-} x_{r,i,j}^t \leq n_j^{\max} h_j^t \qquad \forall\ j \in OPD, t \in T_j, \tag{29}$$

$$\sum_{t \in T_j} h_j^t = l_j d_j \qquad \forall\ j \in OPD, \tag{30}$$

$$h_j^t \in \{0,1\} \qquad \forall\ j \in OPD, t \in T_j, \tag{31}$$

$$l_j \in \{0,1\} \qquad \forall\ j \in OPD. \tag{32}$$

Constraints (29) state that at any time for which task j is assigned, it must be performed by the appropriate number resources. If the binary decision variable $h_j^t = 1$, then task j must be performed during time slice t. It is for this reason that we require $n_j^{\min} \geq 1$. However, if $h_j^t = 0$, then the value of $n_j^{\min}$ is ignored. Constraints (30) state that if task j is performed (i.e., when $l_j = 1$) then it must be performed for exactly d_j time slices. Like RPD tasks, the assigned time slices may be disjoint. The binary decision variables l_j determine whether or not task j is performed at all.

5.9 OPP Tasks

We define $OPP \subseteq M^{\text{pre}}$ to be the set of optional preemptive tasks that may be performed for a partial duration. The unique feature of OPP tasks is that they allow opportunistic resources to perform the task for less than d_j time slices. Thus, we may consider d_j to represent the *maximum* number of time slices in which the task may be performed. Like RPD and OPD tasks, the assigned time slices for OPP tasks may be disjoint. However, unlike OPD tasks, it is implied that $n_j^{\min} = 0$ for all $j \in OPP$.

As an optional task, there is no need for an 'infinite resource' to maintain mathematical feasibility, and therefore no penalty for failing to perform OPP tasks. OPP tasks may be modeled as follows:

$$\sum_{r \in R_j} \sum_{i \in \Delta_{r,j}^-} x_{r,i,j}^t \leq n_j^{\max} h_j^t \qquad \forall\ j \in OPP, t \in T_j, \tag{33}$$

$$\sum_{t \in T_j} h_j^t \leq d_j \qquad \forall\ j \in OPP, \tag{34}$$

$$h_j^t \in \{0,1\} \qquad \forall\ j \in OPP, t \in T_j. \tag{35}$$

6 Payload Delivery

The non-preemptive task types defined in the previous section may also be used in conjunction with payload delivery missions, where payload refers to an expendable resource such as bombs, missiles, or supplies. In this case, it might be helpful to consider each task to be a *target* or *customer*. Let $P \subseteq M^{\text{non}}$ represent the set of targets for which payload delivery is required. Note that each target in set P must be attributed to one of the non-preemptive tasks defined above. The actual task type selected would depend on the nature of the mission. For example, if it is required that all payload delivered to target $j \in P$ must be delivered at the same time by multiple resources, then target j should be classified as an RNS task. Or, if a coordinated strike mission on multiple targets is required, each target that must receive payload should be a member of the same $RNM(s)$ set. Note that *preemptive* tasks may not be included in set P because they allow resources to revisit tasks, thus making it impossible to account for the actual quantity of payload delivered to a target by each resource.

Each task (target) $j \in P$ requires between $k_j^{\min}$ and $k_j^{\max}$ units of payload to be delivered. Let c'_r represent the remaining payload capacity of resource $r \in R$ at the time of the pop-up event, t_0. Let $q_{r,j}$ be the integer decision variable describing the payload expended by resource r at task j.

Again, to maintain a mathematically feasible solution, we define w_j to be the 'infinite' payload capacity. The following constraints capture payload delivery operations:

$$\sum_{j \in \{P \cap M_r\}} q_{r,j} \leq c'_r \qquad \forall\ r \in R, \tag{36}$$

$$q_{r,j} \leq \min(c'_r, k_j^{\max}) \sum_{i \in \Delta^-_{r,j}} \sum_{t \in T_j} x^t_{r,i,j} \qquad \forall\ j \in P, r \in R_j, \tag{37}$$

$$k_j^{\min} \leq \sum_{r \in R_j} q_{r,j} + w_j \leq k_j^{\max} \qquad \forall\ j \in P, \tag{38}$$

$$q_{r,j} \in \{0, 1, \ldots, \min(c'_r, k_j^{\max})\} \qquad \forall\ j \in P, r \in R_j, \tag{39}$$

$$w_j \in \{0, 1, \ldots, k_j^{\min}\} \qquad \forall\ j \in P. \tag{40}$$

Constraints (36) ensure that the amount of payload delivered by resource r does not exceed its capacity. Constraints (37) state that a resource may only deliver payload to a task (target) that it performs. Note that, because we restrict all payload tasks to be attributed to a *non-preemptive* task type, $\sum_{i \in \Delta^-_{r,j}} \sum_{t \in T_j} x^t_{r,i,j} \leq 1$ for all $r \in R$, $j \in P$ (resource r may not re-visit a given non-preemptive task). Constraints (38) ensure that the appropriate units of payload are delivered to task (target) j. Finally, constraints (39) and (40) describe the integer requirements of the decision variables. In another context,

these constraints would also work for payload *pickup*. It is straightforward to see how multiple payload types could be incorporated in the model.

7 Network Constraints

In this section we state the mathematical formulation of the constraints that govern the structure of the underlying network.

7.1 Prohibit Multitasking and Force Initial Location

First, each resource may be assigned to no more than one *node* during any given time slice:

$$\sum_{j \in \{\Delta_r^+ : t \in T_j\}} \sum_{i \in \Delta_{r,j}^-} x_{r,i,j}^t \leq 1 \qquad \forall\ r \in R, t \in T. \tag{41}$$

Next, each resource must begin its route by departing from its initial location, Δ_r^0:

$$\sum_{j \in \{\Delta_r^+ : \Delta_r^0 \in \Delta_{r,j}^-\}} \sum_{t \in T_j} x_{r,\Delta_r^0,j}^t = 1 \qquad \forall\ r \in R. \tag{42}$$

Constraints (42) rely on the assumption that each resource must end at a base location. Consequently, each resource must leave its starting location. In the event that resource r does not have a sufficient amount of fuel remaining to travel to the nearest available base location, the set Δ_r^+ contains the 'dummy' base node, Δ_r^*, which may be reached by resource r from any location in exactly one time slice. Thus, constraints (42) will always be satisfied.

The following constraints ensure that each resource's first assigned task is at a feasible time.

$$\sum_{t \in \{T_j : t < t_0 + f_{r,\Delta_r^0,j}\}} x_{r,\Delta^0,j}^t = 0 \qquad \forall\ r \in R, j \in \{\Delta_r^+ : \Delta_r^0 \in \Delta_{r,j}^-\} \tag{43}$$

Constraints (43) indicate that resource r cannot be assigned to node j before $f_{r,\Delta_r^0,j}$ time slices have passed. These constraints could be used as a pre-processing step, thus eliminating some decision variables.

7.2 Node Predecessors and Successors

Due to the flexibility afforded by preemptive tasks, simple flow balance equations (rate in equals rate out) are not sufficient to form a route. In a feasible route/schedule, each node must be linked to a predecessor

node:

$$\sum_{i \in \{\Delta^-_{r,j}: i \neq k \text{ if } k \notin \Delta^-_{r,k}\}} \sum_{t_j \in \{T_j : t_j + f_{r,j,k} \leq t_k\}} x^{t_j}_{r,i,j} \geq x^{t_k}_{r,j,k} \qquad \forall\, r \in R, k \in \Delta^+_r, j \in \{\Delta^-_{r,k} \setminus \Delta^0_r\}, t_k \in T_k. \quad (44)$$

Constraints (44) indicate that if resource r travels from node j to node k, then it must have previously traveled from another node to j. Because node j appears as a successor node in the left-hand side, we include the condition that j cannot be a source location ($\Delta^-_{r,k} \setminus \Delta^0_r$), as there are no arcs leading *to* Δ^0_r. The condition $i \neq k$ if $k \notin \Delta^-_{r,k}$ ensures that we do not allow the resource to take a subtour from k to j and back to k, *unless* node k is a preemptive task.

We also require that a resource may be assigned to no more than one immediate successor at a given node. For non-preemptive tasks and the initial (source) location node, these constraints are given by

$$\sum_{j \in \{\Delta^+_r : i \in \Delta^-_{r,j}\}} \sum_{t \in T_j} x^t_{r,i,j} \leq 1 \qquad \forall\, r \in R, i \in \{\Delta^0_r \cup (\Delta^+_r \cap M^{\text{non}})\}. \quad (45)$$

In constraints (45), the condition $i \in \{\Delta^0_r \cup (\Delta^+_r \cap M^{\text{non}}\}$ ensures that only nodes that cannot be revisited are considered. The condition $j \in \{\Delta^+_r : i \in \Delta^-_{r,j}\}$ ensures that node j is a node that can be reached directly from node i (i.e., only defined arcs are considered).

Preemptive tasks are unique because the network structure allows a resource to revisit these nodes multiple times. The constraints necessary to prevent a resource from 'splitting' (traveling to two different nodes simultaneously) on a preemptive task are given by

$$\sum_{\substack{t_j \in T_j: \\ t_j \leq t^{\max}}} \sum_{i \in \Delta^-_{r,j}} x^{t_j}_{r,i,j} \geq \sum_{\substack{k \in \Delta^+_r: \\ j \in \Delta^-_{r,k}}} \sum_{\substack{t_k \in T_k: \\ t_k \leq t^{\max} + f_{r,j,k}}} x^{t_k}_{r,j,k} \quad \forall\, r \in R, j \in \{\Delta^+_r \cap M^{\text{pre}}\}, t^{\max} \in T. \quad (46)$$

The left-hand side of constraints (46) represent the number of times that resource r travels *to* preemptive task j by time slice $t^{\max}$, while the right-hand side represents the number of times the resource travels *from* preemptive task j by time $t^{\max}$. In other words, if resource r leaves a preemptive task, it must have previously entered that task at least as many times. Note that $t^{\max} + f_{r,j,k}$ represents the latest possible time at which the resource may leave task j to arrive at task k at time t_k. There are no constraints regarding the successor of a base node, since arcs *leaving* a base are undefined.

7.3 Final Location and Fuel Capacity

Each resource must terminate its route at a base location, even if a resource is not assigned to any tasks. Due to fuel capacity considerations, resources must arrive at the base location before running out of fuel. There are two cases to consider. If resource r is airborne at the time of the pop-up event (i.e., $B'_r = \varnothing$), then it must visit a base location by time slice $t_0 + g'_r$, or terminate at the dummy base node Δ^*_r:

$$\sum_{j \in B_r} \sum_{i \in \Delta^-_{r,j}} \sum_{t \in \{T_j : t \leq t_0 + g'_r\}} x^t_{r,i,j} + \sum_{i \in \Delta^-_{r,\Delta^*_r}} x^{t_0+g'_r+1}_{r,i,\Delta^*_r} = 1 \qquad \forall\, r \in \{R : B'_r = \varnothing\}. \tag{47}$$

However, if r is at a base location at the time of the pop-up (i.e., $B'_r \neq \varnothing$), then it does not begin to consume fuel until it leaves a base. The following constraints ensure that such a resource will either visit a base or terminate at a dummy base within g'_r time slices of the time at which it departed from its initial location:

$$t_0 + 1 \leq \sum_{j \in B_r} \sum_{i \in \Delta^-_{r,j}} \sum_{t \in T_j} t x^t_{r,i,j} + \sum_{i \in \Delta^-_{r,\Delta^*_r}} (t_0 + g'_r) x^{t_0+g'_r+1}_{r,i,\Delta^*_r} \leq \sum_{j \in \Delta^+_r} \sum_{t \in T_j} (t - f_{r,\Delta^0_r,j} + g'_r) x^t_{r,\Delta^0_r,j}$$
$$\forall\, r \in \{R : B'_r \neq \varnothing\}. \tag{48}$$

Constraints (48) rely on the assumption that a resource departing from an initial location corresponding to a base will schedule its departure such that there is no loitering before arriving at the first assigned node. Recall that the network structure requires the resource to depart from its initial location, even if that location is at the same coordinates as a base.

7.4 Special Case for Non-preemptive Tasks

If all tasks are non-preemptive (i.e., if $M = M^{\text{non}}$), then the network structure may be characterized by fewer network constraints. Specifically, constraints (41) – (43), (47), and (48) remain necessary, but constraints (44) – (46) regarding node predecessors and successors may be replaced by the following:

$$\sum_{i \in \Delta^-_{r,j}} \sum_{t \in T_j} x^t_{r,i,j} = \sum_{k \in \{\Delta^+_r : j \in \Delta^-_{r,k}\}} \sum_{t \in T_k} x^t_{r,j,k} \qquad \forall\, r \in R, j \in M^{\text{non}}, \tag{49}$$

$$\sum_{i \in \Delta^-_{r,j}} \sum_{t \in T_j} t x^t_{r,i,j} \leq \sum_{k \in \{\Delta^+_r : j \in \Delta^-_{r,k}\}} \sum_{t \in T_k} (t - f_{r,j,k}) x^t_{r,j,k} \qquad \forall\, r \in R, j \in M^{\text{non}}. \tag{50}$$

Constraints (49) state that a resource must *visit* task j the same number of times as it *leaves* task j. This type of constraint is common in the VRPTW literature. Constraints (50) ensure that if resource r travels from node i to task j to node k, then the resource must travel from i to j *before* traveling from j to k. That is, the time at which task k begins must not be prior to the time at which task j begins plus the travel time from j to k. This constraint is not found in VRPTW formulations, because we incorporate the time slice in the binary decision variable.

It should be noted that while this "special case" formulation requires fewer constraints to guarantee a valid route for each resource, if the integer requirements on the binary decision variables are relaxed the resulting solution may not be as tight as it is for the "full" formulation. Thus, a trade-off exists between the reduction in time required to construct the constraints and an increase in time required to solve the ILP via off-the-shelf solvers such as CPLEX. However, this "special case" formulation may prove useful for checking feasibility in future heuristic solution approaches.

8 Objective Function Terms

We consider four objectives when solving the DRM problem. First, we seek to maximize overall mission effectiveness. Second, we would like to adhere to the initial mission plan as much as possible by minimizing the changes to the initial assignments. Third, whenever changes must occur, we would like to minimize the total travel time of all resources. Finally, we would like to minimize the number of 'infinite resources' that are utilized. Recall that these fictitious resources are included in the model to guarantee mathematical feasibility. In this section we state the mathematical representation of these objectives.

8.1 Maximize Overall Mission Effectiveness

First, we look to maximize the overall mission effectiveness, while considering potential time preferences within each task's allowable time window. Let Z_E represent the objective function term that captures overall mission effectiveness:

$$Z_E \equiv \sum_{j \in M} \sum_{r \in R_j} \sum_{t \in T_j} \sum_{i \in \Delta^-_{r,j}} p_j e_{r,j} \frac{1}{d'_j} \left(1 - \xi_j \left|\tau_j - t\right|\right) x^t_{r,i,j} + \sum_{j \in P} \sum_{r \in R_j} p_j q_{r,j}.$$

We scale Z_E by d'_j, as defined in equation (2), to prevent preemptive tasks from being unduly weighted. The parameter τ_j represents the preferred time for task $j \in M$ to be performed, such that $\min(T_j) \leq \tau_j \leq \max(T_j)$. Note that τ_j may be a real number, and as such is not restricted to being stated in terms of integer time slices. For example, if $T_j = \{4, 5, 6, 7\}$ and it is desirable to perform task j in

the middle of this time window, then $\tau_j = 5.5$. The scaling parameter ξ_j should be defined such that $0 < (1 - \xi_j |\tau_j - t|) \leq 1$ for all $t \in T_j$, where $|\tau_j - t|$ denotes the absolute value of the difference between the preferred time for task j and the possible assigned time slice, t. If no preferred time slice exists for task $j \in M$, then ξ_j should equal zero. One valid definition of ξ_j is given by the following, although other definitions are acceptable:

$$\xi_j = \begin{cases} \min\left(\frac{1}{\tau_j - \min(T_j)+1}, \frac{1}{\max(T_j) - \tau_j + 1}\right) & \text{if } \tau_j \text{ exists,} \\ 0 & \text{otherwise.} \end{cases}$$

Since $0 < (1 - \xi_j |\tau_j - t|) \leq 1$, it follows that $0 < p_j e_{r,j} (1 - \xi_j |\tau_j - t|) \leq p_j e_{r,j}$ for all $t \in T_j$. If payload delivery is required (constraints (36) – (40)), the second set of summations reflects a priority-based incentive for delivering payload.

The upper bound for Z_E, $Z_E^{\max}$, is given by:

$$Z_E \leq Z_E^{\max} \equiv \sum_{j \in M} \left(p_j \max_{r \in R_j} (e_{r,j})\, n_j^{\max} + p_j k_j^{\max} \right).$$

8.2 Minimize Changes to Initial Assignments

Our second objective is to minimize changes to the initial mission plan. We define three types of changes. The first type involves changes in assigned task times. The second involves changes in the order of assigned tasks. Finally, change may occur when a task is performed by a different resource.

We begin by defining the following parameters that will simplify our notation:

$$a_{r,i,j} \equiv \min\left(1, \sum_{t \in T_j} a^t_{r,i,j}\right) \qquad \forall\ r \in R, j \in \Delta_r^+, i \in \Delta_{r,j}^-.$$

In a similar fashion we may define $a^t_{r,j}$ and $a_{r,j}$. Thus, $a_{r,i,j} = 1$ if resource r was initially assigned to travel from node i to node j at any time, $a^t_{r,j} = 1$ if resource r was initially assigned to perform task j at time t (regardless of previous task), and $a_{r,j} = 1$ if resource r was initially assigned to perform task j (regardless of previous task or time).

8.2.1 Change in Order

Let Z_{CO} represent the number of times the *order* of all resources' assigned tasks are changed from the initial plan. Then

$$Z_{CO} \equiv \sum_{r \in R} \sum_{j \in \Delta_r^+} \sum_{\substack{i \in \Delta_{r,j}^- \\ i \neq j}} \sum_{t \in T_j} (1 - a_{r,i,j})\, x_{r,i,j}^t.$$

Note that changing the assigned *time slices* does not lead to a penalty in the above terms. Also, there is no penalty for failing to perform an initially assigned task. The condition '$i \neq j$' is included for the following reason: suppose the initial plan did not assign task r to *preemptive* task j. Then we do not care if r now continues to perform task j. We only want to capture the arc from another node to task j. Note that $j \notin \Delta_{r,j}^-$ if $j \in M^{\text{non}}$.

To calculate the upper bound for Z_{CO}, $Z_{CO}^{\max}$, we first note that at most $n_j^{\max}$ resources may perform task j. Each resource now assigned to task j may visit task j at most d_j' times. So, for each task j, $n_j^{\max} d_j'$ represents the case where each resource assigned to task j takes a different arc to task j. Finally, there are $|R|$ possible new arcs that the resources could take to a base location.

$$Z_{CO} \leq Z_{CO}^{\max} \equiv \sum_{j \in M} \left(n_j^{\max} d_j'\right) + |R|.$$

8.2.2 Change in Time

Now, Z_{CT} represents the objective function term that captures changes in the assigned task times:

$$Z_{CT} \equiv \sum_{r \in R} \sum_{j \in \Delta_r^+} \sum_{i \in \Delta_{r,j}^-} \frac{a_{r,i,j}}{d_j'} \sum_{t \in T_j} \left(1 - a_{r,j}^t\right) x_{r,i,j}^t.$$

This term is scaled by d_j' to prevent preemptive tasks from being excessively penalized, since these tasks are assigned on a slice-by-slice basis. Note that no penalty is imposed for changing the *order* of tasks visited by resource r, by virtue of using the binary $a_{r,i,j}$ parameter, which equals one only if resource r was initially assigned to travel from node i to node j. Also, there is no penalty for failing to perform an initially assigned task.

The upper bound for Z_{CT}, $Z_{CT}^{\max}$, is given by:

$$Z_{CT} \leq Z_{CT}^{\max} \equiv \sum_{j \in M} n_j^{\max} + |R|.$$

The summation captures the fact that each task j could be performed at a new time, and each task could be assigned to at most $n_j^{\max}$ resources. Since each resouce could be assigned to a base at a new time, we also add $|R|$, the number of resources.

8.2.3 Change in Resource

Finally, we impose a penalty for failing to perform an initially-assigned task with the same resource. For *non-preemptive* tasks, this penalty is given by

$$Z_{CR}^{\text{non}} \equiv \sum_{j \in M^{\text{non}}} \sum_{r \in R_j} a_{r,j} \left(1 - \sum_{i \in \Delta_{r,j}^-} \sum_{t \in T_j} x_{r,i,j}^t \right).$$

Similarly, the penalty for a resource visiting a different base may be expressed as:

$$Z_{CR}^{\text{base}} \equiv \sum_{r \in R} \sum_{j \in B_r} a_{r,j} \left(1 - \sum_{i \in \Delta_{r,j}^-} \sum_{t \in T_j} x_{r,i,j}^t \right).$$

For *preemptive* tasks, the procedure for determining a change in resource requires the inclusion of a new binary decision variable. Let $x_{r,j}^{\text{pre}} = 1$ if the new mission plan assigns resource r to preemptive task j, regardless of predecessor task or assigned time ($x_{r,j}^{\text{pre}} = 0$ otherwise). This binary decision variable is required because resources are permitted to re-visit preemptive tasks. As such, $\sum_{i \in \Delta_{r,j}^-} \sum_{t \in T_j} x_{r,i,j}^t$ may be greater than one for task $j \in M^{\text{pre}}$. The following constraints force $x_{r,j}^{\text{pre}}$ to assume the correct value:

$$x_{r,j}^{\text{pre}} \leq \sum_{i \in \Delta_{r,j}^-} \sum_{t \in T_j} x_{r,i,j}^t \leq d_j x_{r,j}^{\text{pre}} \qquad \forall\ j \in M^{\text{pre}}, r \in R_j,$$

$$x_{r,j}^{\text{pre}} \in \{0,1\} \qquad \forall\ j \in M^{\text{pre}}, r \in R_j.$$

Thus, we may write the term capturing the change in resources for *preemptive* tasks as:

$$Z_{CR}^{\text{pre}} \equiv \sum_{j \in M^{\text{pre}}} \sum_{r \in R_j} a_{r,j} \left(1 - x_{r,j}^{\text{pre}} \right).$$

Therefore, the expression for the change in resource is given by

$$Z_{CR} \equiv Z_{CR}^{\text{non}} + Z_{CR}^{\text{pre}} + Z_{CR}^{\text{base}}.$$

There is no penalty for a change in time or a change in order.

Using the fact that each task could have at most $n_j^{\max}$ different resources assigned to it, and each of the $|R|$ resources could terminate its route at a different base, the upper bound for Z_{CR} is given by

$$Z_{CR} \leq Z_{CR}^{\max} \equiv \sum_{j \in M} n_j^{\max} + |R|.$$

8.3 Minimize Travel Time

The total travel time for all resources, Z_T, is given by:

$$Z_T \equiv \sum_{r \in R} \sum_{j \in \Delta_r^+} \sum_{i \in \Delta_{r,j}^-} \sum_{t \in T_j} f_{r,i,j} x_{r,i,j}^t.$$

An upper bound on the total travel time, $Z_T^{\max}$, captures the fact that each resource must end at a base prior to running out of fuel (where the maximum remaining endurance of resource r is given by g_r').

$$Z_T \leq Z_T^{\max} \equiv \sum_{r \in R} g_r'.$$

8.4 Minimize 'Infinite Resources'

Although 'infinite resources' may be necessary to guarantee mathematical feasibility, their use should be minimized. We have defined three types of 'infinite resources' in our model. The first type is used in the constraints involving required (mission-critical) tasks. Let Z_{IR} represent the penalty for using these 'infinite resources':

$$Z_{IR} \equiv \sum_{j \in \left(RNA \cup RNP \cup \{RPD : n_j^{\max} = 1\}\right)} \frac{p_j}{d_j'} y_j + \sum_{j \in \left(RNS \cup RNM(s) \cup RPC \cup \{RPD : n_j^{\max} > 1\}\right)} \sum_{t \in T_j} \frac{p_j}{d_j'} z_j^t.$$

We multiply each term by p_j to capture the relative importance of each task and divide by d_j' to avoid over-weighting preemptive tasks. Note that optional tasks do not require the use of an infinite resource.

The second type of 'infinite resource' involves the 'dummy' bases that may be required for resources that do not have sufficient fuel capacity to travel from their current location to the nearest available base. Let Z_{DB} represent the number of resources that must use their 'dummy' base:

$$Z_{DB} \equiv \sum_{r \in R} \sum_{i \in \Delta_{r,\Delta_r^*}^-} x_{r,i,\Delta_r^*}^{t_0 + g_r' + 1}.$$

Finally, let Z_{IP} represent the penalty imposed for using 'infinite payload' capacity:

$$Z_{IP} \equiv \sum_{j \in P} p_j w_j.$$

We multiply each term by p_j to capture the relative importance of each task (target).

8.5 Objective Function Formulation

Now that we have defined all of the individual objective function terms, we must combine these terms into one unified objective function. We do so with the aid of two scaling parameters, α and β, where $0 \leq \alpha \leq 1$ and $0 \leq \beta \leq 1$. Scaling parameter α represents the trade-off between maximizing overall mission effectiveness and minimizing changes to the initial mission plan. The second scaling parameter, β, operates independently and allows the decision maker to specify the importance that should be given to minimizing total travel time. In an effort to maintain a relatively even weighting of the individual objective function terms, we scale each term by its maximum value, as defined above. The lone exception to this being the terms for 'infinite resource' penalties, which we would like to make relatively strong.

Our proposed objective function may therefore be written as follows:

$$\text{Max } \alpha \frac{Z_E}{Z_E^{\max}} - (1-\alpha)\left(\frac{Z_{CO}}{Z_{CO}^{\max}} + \frac{Z_{CT}}{Z_{CT}^{\max}} + \frac{Z_{CR}}{Z_{CR}^{\max}}\right) - \beta \frac{Z_T}{Z_T^{\max}} - Z_E^{\max}\left(Z_{IR} + Z_{DB} + Z_{IP}\right).$$

This function should be optimized, subject to constraints (1) – (50).

8.6 Generating Initial Mission Plans

Although our model is motivated by re-routing problems, it is also capable of generating an off-line initial mission plan. To use the model in this manner, simply set $\alpha = 0$. This indicates that there is no initial mission plan available and accomodates that fact that the $a_{r,i,j}^t$ parameters are undefined. The set of tasks, M, would not include any pop-up tasks.

9 Numerical Example

In this section we provide an example that demonstrates the application of our extensible modeling framework in the context of intelligence, surveillance, and reconnaissance (ISR) missions. This example is characterized by a pop-up threat, payload capacity requirements, BDA activities, and multiple-resource coordination via the RNM task type.

Consider an ISR and strike mission that requires the use of one MQ-1 (Predator) and two RQ-4 (Global Hawk) UAVs. The Predator, named P1, is equipped with two AGM-114 (Hellfire) laser-guided missiles and a rudimentary camera capable of taking snapshot photographs. Both Global Hawks, named GH1 and GH2, are fitted with image recording equipment and a laser projection device that may be used to direct the Predator's laser-guided missiles.

The Predator's primary role is to destroy two enemy targets by firing one laser-guided missile at each target, although it is also assigned to capture still images at two other locations. GH1 is initially assigned to four ISR activities that require the UAV to capture either video or still images at four distinct locations. Each video task demands 30 minutes of uninterrupted target surveillance, while the still images may be captured instantaneously as GH1 passes over the waypoint locations. GH2 is primarily responsible for conducting a BDA of the two targets destroyed by P1. Each BDA task is 40 minutes in duration, and begins 20 minutes after P1 has launched the missile at the target. GH2 is also tasked with taking instantaneous snapshot photographs at two other locations.

Now, suppose that a new threat is identified 50 minutes after the mission commences, as the UAVs are en route to their first assigned task. This pop-up threat is determined to be a high-priority target that the mission commander would like to destroy with a Hellfire missile. Because this target is in a populated area, one of the Global Hawk UAVs must laser-illuminate the target while the Predator launches one laser-guided missile, thus increasing the accuracy of the strike. If this coordinated attack is assigned in the updated mission plan, a 40-minute BDA is required, beginning at least 20 minutes after the missile is launched. Figure 4 shows the location of the pop-up threat, and includes the UAV paths contained in the initial mission plan as well as allowable time windows for all tasks. All times are expressed in units of 'minutes elapsed since the start of the mission.' Nodes 16, 17, and 18 represent the current location of GH1, GH2, and P1, respectively. Node 19 represents the base location where each UAV must terminate its mission before running out of fuel. All other nodes represent one of the tasks described above. Symbols used in the figures for this example are shown in Figure 5.

We now demonstrate how an optimal updated mission plan may be determined. For this example, we assume time slices are 10 minutes in duration. Table 3 contains the data inputs required to characterize each resource at the time of the pop-up event. Each resource has consumed 50-minutes-worth of fuel, such that GH1 and GH2 may remain airborne for 2110 minutes (211 time slices) and P1 may remain airborne for 1390 minutes (139 time slices). The predator is the only UAV with missile payload capacity, and it has its full complement of two missiles still on board.

Nodes 1 – 15 in the network correspond to the 15 tasks that have yet to be completed, as seen in

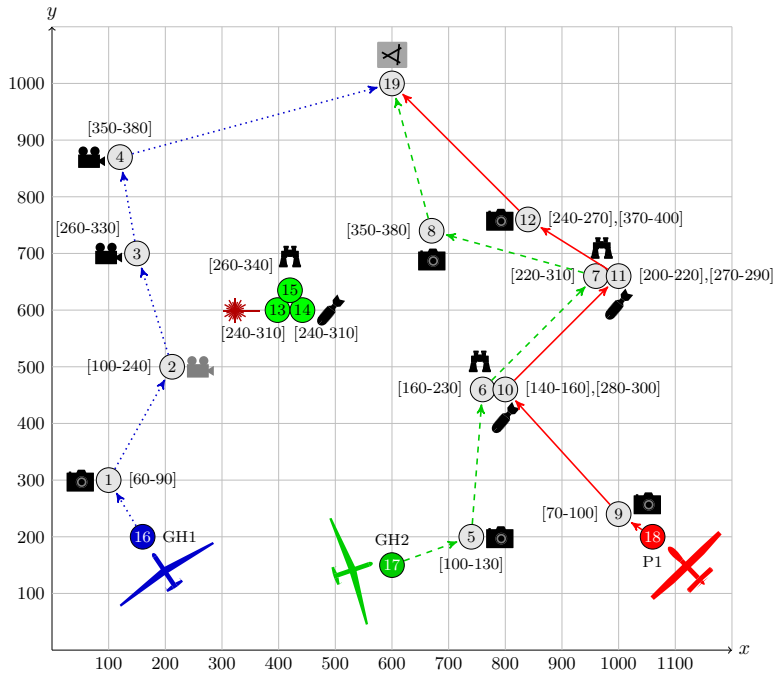

Figure 4: Popup target appears. Initial UAV paths are shown.

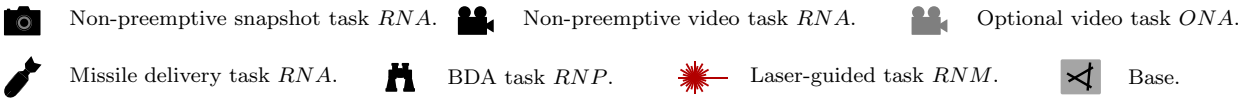

Figure 5: Key for example figures.

Table 4. Tasks 13 – 15 are the tasks associated with the pop-up threat. For this example, all tasks are modeled as non-preemptive, and each of the required tasks must be performed by exactly one resource. The location of each task is specified in Cartesian (x, y) coordinates, measured in miles from a reference point located at $(0, 0)$, although latitude/longitude coordinates could also be used. The base, uniquely identified in the network as node 19, is located at $(600, 1000)$. Priority values for each task, p_j, have been supplied by the mission commander, based on a mapping of high/medium/low to values 10/5/1. The set of allowable starting times for each task, T_j, is also assumed to have been given by the mission commander, and is expressed in terms of 10-minute time slices. For example, $T_{15} = [26, 34]$ implies that task 15 must begin no earlier than 260 minutes after the mission started, and may begin no later than 340 minutes

Table 3: Resource status.

Resource, r	Δ_r^0	Name	Type	Current Location (x, y)	Velocity [mph]	g'_r	c'_r
1	16	GH1	Global Hawk	(160,200)	403	211	0
2	17	GH2	Global Hawk	(600,150)	403	211	0
3	18	P1	Predator	(1060,200)	253	139	2

Table 4: Task details.

Task, j	Task Type	Description	d_j	p_j	(x, y)	T_j	$n_j^{\min}$	$n_j^{\max}$
1	RNA	Snapshot	0	5 (med)	(100,300)	[6-9]	1	1
2	ONA	Optional Video	3	1 (low)	(212,500)	[10-24]	0	1
3	RNA	Video	3	1 (low)	(150,700)	[26-33]	1	1
4	RNA	Video	3	1 (low)	(120,870)	[35-38]	1	1
5	RNA	Snapshot	0	1 (low)	(740,200)	[10-13]	1	1
6	RNP	BDA of Task 10	4	5 (med)	(760,460)	[16-23]	1	1
7	RNP	BDA of Task 11	4	5 (med)	(960,660)	[22-31]	1	1
8	RNA	Snapshot	3	5 (med)	(670,740)	[35-38]	1	1
9	RNA	Snapshot	0	5 (med)	(1000,240)	[7-10]	1	1
10	RNA	Missile Launch	0	5 (med)	(800,460)	[14-16],[28-30]	1	1
11	RNA	Missile Launch	0	5 (med)	(1000,660)	[20-22],[27-29]	1	1
12	RNA	Snapshot	0	5 (med)	(840,760)	[24-27],[37-40]	1	1
13	RNM	Laser Guide	0	10 (high)	(400,600)	[24-31]	1	1
14	RNM	Missile Launch	0	10 (high)	(440,600)	[24-31]	1	1
15	RNP	BDA of Task 14	4	5 (med)	(420,630)	[26-34]	1	1

Table 5: Resource effectiveness values, $e_{r,j}$.

	Task, j														
Resource, r	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1 (GH1)	3	3	3	3	3	3	3	3	1	-1	-1	1	3	-1	3
2 (GH2)	3	3	3	3	3	3	3	3	1	-1	-1	1	3	-1	3
3 (P1)	1	1	1	1	1	-1	-1	1	3	3	3	3	-1	3	-1

after the mission started. The mission commander has decided that each task should be performed as early as possible within its allowable time window, such that $\tau_j = \min(T_j)$ for $j = 1, 2, \ldots, 15$.

To account for the differing capabilities of the Global Hawk and Predator aircraft, effectiveness values are included, as per Table 5. For this particular example, the values of $e_{r,j}$ are chosen such that a value of 3 indicates 'good' performance, 1 indicates 'poor' performance, and -1 indicates that the resource is incapable of performing the task. Finally, the initial mission assignments are shown in Table 6.

The optimal updated mission plan is shown in Figure 6, where we assumed Euclidean distance between nodes, constant-velocity flight, and scaling parameters of $\alpha = 0.74$ and $\beta = 0.50$. The corresponding optimal non-zero decision variable values are provided in Table 7. Because this example contains only non-preemptive tasks, the 'special case' version of the network constraints, constraints (49) and (50), may be used. This results in 320 constraints and 1600 decision variables, with a problem generation time of 0.02 seconds and a solution time of 0.16 seconds. Had constraints (44), (45), and (46) been employed instead, a total of 1567 constraints would be required, with a corresponding problem generation time of 0.05 seconds and a solution time of 0.08 seconds. The optimal solution to this example was obtained by CPLEX version 11.2.1 on a 3.00 GHz Intel Core 2 Duo PC running Windows XP.

Table 6: Initial resource assignments.

		Start Time		
Node, j	Resource, r	Minutes Elapsed	Time Slice, t	$a^t_{r,i,j}$
1	1 (GH1)	70	7	$a^{7}_{1,16,1} = 1$
2	1 (GH1)	120	12	$a^{12}_{1,1,2} = 1$
3	1 (GH1)	260	26	$a^{26}_{1,2,3} = 1$
4	1 (GH1)	350	35	$a^{35}_{1,3,4} = 1$
5	2 (GH2)	100	10	$a^{10}_{2,17,5} = 1$
6	2 (GH2)	170	17	$a^{17}_{2,5,6} = 1$
7	2 (GH2)	260	26	$a^{26}_{2,6,7} = 1$
8	2 (GH2)	350	35	$a^{35}_{2,7,8} = 1$
9	3 (P1)	70	7	$a^{7}_{3,18,9} = 1$
10	3 (P1)	150	15	$a^{15}_{3,9,10} = 1$
11	3 (P1)	220	22	$a^{22}_{3,10,11} = 1$
12	3 (P1)	270	27	$a^{27}_{3,11,12} = 1$
19 (Base)	1 (GH1)	2160	216	$a^{216}_{1,4,19} = 1$
19 (Base)	2 (GH2)	2160	216	$a^{216}_{2,8,19} = 1$
19 (Base)	3 (P1)	1440	144	$a^{144}_{3,12,19} = 1$

Table 7: Optimal decision variable values.

Decision Variable	Value	Comments
$x^{7}_{1,16,1}$	1.00	Resource 1 departs from its initial location, beginning Task 1 at time slice 7.
$x^{12}_{1,1,2}$	1.00	
$x^{25}_{1,2,13}$	1.00	
$x^{30}_{1,13,3}$	1.00	
$x^{36}_{1,3,4}$	1.00	
$x^{221}_{1,4,19}$	1.00	Resource 1 arrives at base (during time slice when it runs out of fuel).
$x^{10}_{2,17,5}$	1.00	
$x^{17}_{2,5,6}$	1.00	
$x^{27}_{2,6,15}$	1.00	
$x^{36}_{2,15,8}$	1.00	
$x^{221}_{2,8,19}$	1.00	
$x^{7}_{3,18,9}$	1.00	
$x^{15}_{3,9,10}$	1.00	
$x^{25}_{3,10,14}$	1.00	
$x^{37}_{3,14,12}$	1.00	
$x^{149}_{3,12,19}$	1.00	
w_{11}	1.00	1 unit of required payload *not* delivered to Task 11.
y_{11}	1.00	1 required resource *not* assigned to Task 11.
$q_{3,10}$	1.00	Resource 3 delivered 1 unit of payload to Task 10.
$q_{3,14}$	1.00	Resource 3 delivered 1 unit of payload to Task 14.
$b_{25,1}$	1.00	The RNM tasks for set 1 were performed at time slice 25.
$x^{\mathrm{RNP}'}_{6}$	1.00	The predecessor task for the RNP (BDA) task 6 is assigned.
$x^{\mathrm{RNP}'}_{15}$	1.00	The predecessor task for the RNP (BDA) task 15 is assigned.
$x^{\mathrm{RNP}}_{17,6}$	1.00	RNP task 6 is assigned at time slice 17.
$x^{\mathrm{RNP}}_{27,15}$	1.00	RNP task 15 is assigned at time slice 27.

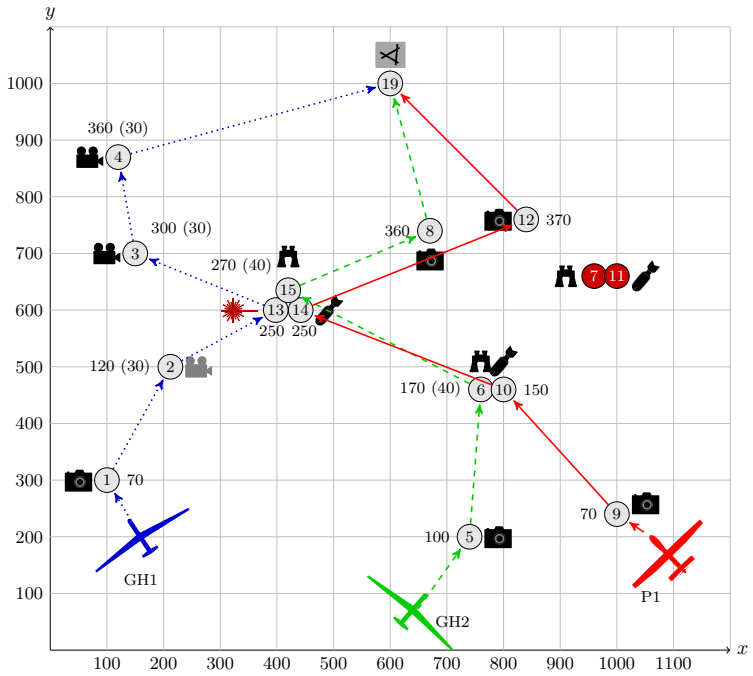

Figure 6: Re-assignments for example problem.

10 Summary and Future Research

We have proposed an extensible modeling framework for the dynamic re-routing of resources in response to changes in battlespace conditions. Nine unique task types were defined which demonstrate the flexibility of this model. These tasks are representative of many of the tasks that are performed by UAVs. Tasks may also require or benefit from multiple resources performing them simultaneously or in sequence. The model also addresses fuel and payload capacity restrictions. Another key feature of the modeling framework is the inclusion of 'infinite resources' and 'dummy' bases, which not only guarantee mathematical feasibility but also warn decision makers where resource shortfalls exist. Because the DRM problem involves the re-allocation of resources that were previously operating under an initial mission plan, our objective function balances the maximization of overall mission effectiveness with the minimization of changes to the original task assignments.

As with all vehicle routing problems, solution times can increase dramatically as problems become larger, thus rendering commercial IP solvers useless for DRM problems, where updated mission plans are required in near-real-time. For this reason, future research should focus on the development of solution approaches that can provide high-quality, if not optimal, solutions to large-scale problems in a rapid fashion. Another area that warrants further investigation is the development of new task types to reflect the ever-growing use of UAVs. Finally, the usefulness of the model may be further enhanced with the addition of resource refueling operations, variable resource velocities, and the imposition of turning radius

restrictions.

Acknowledgements

The authors are grateful for the valuable feedback provided by the two anonymous referees. This research has been funded partially under the following programs:

1. Office of Naval Research under contract number N00173-08-C-4009.
2. Air Force Research Laboratories through Sierra Nevada Corporation.
3. Air Force Research Laboratories through IAVO Corp.

References

[1] J. Bellingham, M. Tillerson, A. Richards, and J. P. How. Multi-task allocation and path planning for cooperating uavs. In *Cooperative Control: Models, Applications and Algorithms*, pages 1–19. Conference on Coordination, Control and Optimization, November 2001.

[2] J. Berger, M. Barkaoui, and A. Boukhtouta. A hybrid genetic approach for airborne sensor vehicle routing in real-time reconnaissance missions. *Aerospace Science and Technology*, 11:317–326, 2007.

[3] L. Bianchi. Notes on dynamic vehicle routing – the state of the art –. Technical Report Technical Report IDSIA-05-01, December 2000. URL `ftp://ftp.idsia.ch/pub/techrep/IDSIA-05-01.ps.gz`.

[4] E. Bone and C. Bolkcom. Unmanned aerial vehicles: Background and issues for congress. Report for Congress Order Code RL31872., April 2003. URL `http://www.fas.org/irp/crs/RL31872.pdf`.

[5] J.F. Cordeau and G. Laporte. A tabu search algorithm for the site dependent vehicle routing problem with time windows. *Infor-Information Systems and Operational Research*, 39(3):292–298, 2001.

[6] M. Cox. Cargo uavs considered for supply delivery. *Army Times*, March 15, March 2009. URL `http://www.armytimes.com/news/2009/03/army_giant_uavs_031409w/`.

[7] Department of Defense. Fy2009-2034 unmanned systems integrated roadmap. 2009. URL `http://www.acq.osd.mil/uas/docs/UMSIntegratedRoadmap2009.pdf`.

[8] D. Favaretto, E. Moretti, and P. Pellegrini. Ant colony system for a vrp with multiple time windows and multiple visits. *Journal of Interdisciplinary Mathematics*, 10(2):263–284, 2007.

[9] T. Flatberg, G. Hasle, O. Kloster, E.J. Nilssen, and A. Riise. Dynamic and stochastic aspects in vehicle routing-a literature survey. Technical report, SINTEF Technical Report STF90A05413, 2005.

[10] A. Goel and V. Gruhn. A general vehicle routing problem. *European Journal of Operational Research*, 191: 650–660, 2008.

[11] Fabrice Janez. Optimization method for sensor planning. *Aerospace Science and Technology*, 11:310–316, 2007. doi: doi:10.1016/j.ast.2006.12.005.

[12] N. Jozefowiez, F. Semet, and E. Talbi. Multi-objective vehicle routing problems. *European Journal of Operational Research*, 189:293–309, 2008.

[13] D.B. Kingston and C.J. Schumacher. Time-dependent cooperative assignment. In *Proceedings of the 2005 American Control Conference*, pages 4084–4089, 2005.

[14] G.W. Kinney, R.R. Hill, and J.T. Moore. Devising a quick-running heuristic for an unmanned aerial vehicle (uav) routing system. *Journal of the Operational Research Society*, 56:776–786, 2005.

[15] H.C. Lau, M. Sim, and K.M. Teo. Vehicle routing problem with time windows and a limited number of vehicles. *European Journal of Operational Research*, 148:559–569, 2003.

[16] J.-Q. Li, P.B. Mirchandani, and D. Borenstein. Real-time vehicle rerouting problems with time windows. *European Journal of Operational Research*, 194:711–727, 2009.

[17] P. Pellegrini, D. Favaretto, and E. Moretti. Multiple Ant Colony Optimization for a Rich Vehicle Routing Problem: A Case Study. *Lecture Notes in Computer Science*, 4693:627–634, 2007.

[18] H.N. Psaraftis. Dynamic vehicle routing: Status and prospects. *Annals of Operations Research*, 61:143–164, 1995.

[19] C. Schumacher, P.R. Chandler, M. Pachter, and L.S. Pachter. Optimization of air vehicles operations using mixed-integer linear programming. *Journal of the Operational Research Society*, 58(4):516–527, 2007.

[20] V.K. Shetty, M. Sudit, and R. Nagi. Priority-based assignment and routing of a fleet of unmanned combat aerial vehicles. *Computers & Operations Research*, 35:1813–1828, 2008.

[21] T. Shima, S.J. Rasmussen, A.G. Sparks, and K.M. Passino. Multiple task assignments for cooperating uninhabited aerial vehicles using genetic algorithms. *Computers & Operations Research*, 33:3252–3269, 2006.

[22] E. Sofge. Houston cops' test drone now in iraq, operator says. *Popular Mechanics*, November 28, November 2007. URL `http://www.popularmechanics.com/science/air_space/4234272.html`.

[23] M. Tavana, M.D. Bailey, and T.E. Busch. A multi-criteria vehicle-target allocation assessment model for network-centric joint air operations. *International Journal of Operational Research*, 3(3):235–254, 2008.

[24] Paolo Toth and Daniele Vigo, editors. *The Vehicle Routing Problem*. SIAM Monographs on Discrete Mathematics and Applications, Philadelpha, PA, 2002.

[25] P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden. A guided local search metaheuristic for the team orienteering problem. *European Journal of Operational Research*, 196(1):118–127, 2009.

[26] G. Warwick and J. M. Doyle. Predator uav set for u.s.-canada patrol. *Aviationweek.com*, December 5 2008. URL `http://www.aviationweek.com/aw/generic/story_channel.jsp?channel=defense&id=news/PRED12058.xml&headline=Predator%20UAV%20Set%20for%20U.S.-Canada%20Patrol`.

[27] A.L. Weinstein and C. Schumacher. Uav scheduling via the vehicle routing problem with time windows.

Technical Report AFRL-VA-WP-TP-2007-306, Air Force Research Laboratory, January 2007. Conference paper submitted to the Proceedings of the 2007 AIAA Infotech Aerospace Conference and Exhibit.

[28] J. Wilde, D. DiBiaso, and M. Nervegna. Team planning for unmanned vehicles in the risk-aware mixed-initiative dynamic replanning system. *Oceans 2007*, pages 1–8, 2007.