Dynamic Resource Allocation for Efficient Sharing of Services from Heterogeneous Autonomous Vehicles

Bilal Y. Kaddouh[1], William J. Crowther[2], Peter Hollingsworth[3]
The University of Manchester, Manchester, England M13 9PL, UK

A novel dynamic resource allocation model is introduced for efficient sharing of services provided by ad hoc assemblies of heterogeneous autonomous vehicles. A key contribution is the provision of capability to dynamically select sensors and platforms within constraints imposed by time dependencies, refueling, and transportation services. The problem is modeled as a connected network of nodes and formulated as an Integer Linear Program (ILP). Solution fitness is prioritized over computation time. Simulation results of an illustrative scenario are used to demonstrate the ability of the model to plan for sensor selection, refueling, collaboration and cooperation between heterogeneous resources. Prioritization of operational cost leads to missions that use cheaper resources but take longer to complete. Prioritization of completion time leads to shorter missions at the expense of increased overall resource cost. Missions can be successfully re-planned through dynamic reallocation of new requests during a mission. Monte Carlo studies on systems of increasing complexity show that good solutions can be obtained using low time resolutions, with small time windows at a relatively low computational cost. In comparison with other approaches, the developed ILP model provides best solutions at the expense of longer computation time.

Nomenclature

α	=	summation of difference between gain and maintenance cost, £
β	=	summation of landing cost of all resources, £
γ	=	summation of take-off cost of all resources, £
δ	=	summation of traveling time of all resources, time slices
ε	=	summation of finishing time of all resources, time slices
θ	=	weighted mission completion time, time slices
ι	=	summation of remaining fuel time in all resources, time slices
κ	=	summation of resources carried payloads
λ	=	summation of number of time slices allocated for loiter tasks
$AOC_{r,b}$	=	additional operational cost of operating resource r from base b, £
B_r	=	set of bases that can be visited by resource r
$BASES$	=	set of all bases
CAP_r	=	payload capacity of resource r
$Comptime$	=	overall completion time of the mission, time slices
c	=	very small number(<0.01)
c_1	=	weight of effectiveness
c_2	=	weight of operations cost
c_3	=	weight of take-off and landing costs
c_4	=	weight of individual completion time
c_5	=	weight of mission completion time

[1] Doctoral candidate, School of Mechanical, Aerospace and Civil Engineering, George Begg Building, Manchester, UK, M13 9PL, bilal.kaddouh@manchester.ac.uk, Member AIAA
[2] Reader in Engineering, School of Mechanical, Aerospace and Civil Engineering, George Begg Building, Manchester, UK, M13 9PL, w.j.crowther@manchester.ac.uk.
[3] Lecturer in Aerospace, School of Mechanical, Aerospace and Civil Engineering, George Begg Building, Manchester, UK, M13 9PL, peter.hollingsworth@manchester.ac.uk, Senior Member AIAA

d_j = binary decision variable, used to indicate if the payload to be delivered at j has been picked up
$dt_{j,t}$ = binary decision variable used to store the pickup time of the payload to be delivered at j
$DELIVER$ = delivery subtask of a transport task
$From_{r,j}$ = set of all nodes from which resource r can travel to node j
$Fintime_r$ = integer variable representing the finishing time slice of each resource r
G_j = gain obtained from performing task j, £
$InitPos_r$ = set of initial positions for resource r
$InitSen_r$ = set of initial sensor combination onboard resource r
$INTER$ = set of intersection tasks
l_j = number of time slices required to complete task j
lag_j = lag time between preceded task j and its preceding task, time slices
$LC_{r,b}$ = cost of landing resource r at base b
$LOITER$ = set of all loiter tasks
MC_r = maintenance cost per time slice of operation of resource r, £/time slices
$NPre$ = set of non-preemptive nodes
NR = number of resources used in the simulation studies
NT = number of tasks used in the simulation studies
$Ntog$ = number of task sets that need to be completed together
NTS = number of time slices used for planning
$PE_{r,j}$ = platform effectiveness of resource r in performing task j
$Pick_j$ = pickup subtask associated with deliver task j
$PotINTER$ = tasks that can potentially have an *Intersection* task
Pre = set of preemptive nodes
$PRECEDED$= set of tasks that cannot start execution before at least one preceding tasks have been completed
$PRECEDING_j$= set containing the tasks that precede task j
PW_j = payload weight of deliver task j
px_j = binary decision variable used to indicate if any preceding task j has been allocated
$pxt_{j,t}$ = binary decision variable used to store the time t at which the preceding task of j has been allocated
Q = maximum fuel capacity of all resources, time slices
Q_r = maximum fuel capacity of resource r, time slices
R_j = set of resources that have the capability to attend node j
$RESOURCES$ = set of all vehicles
S_r = set of sensor combinations that can be installed on resource $r \in RESOURCES$
$SE_{s,j}$ = effectiveness of sensor s in performing task j
$SENSORS$= set of all sensor combinations
$SR_{r,s}$ = number of time slices reduced due to the installation of sensor s onboard resource r
SUC_s = cost per time slice of operating sensor s, £/time slices
T_b = set of time slices when base b is available
T_j = set of time slices when task j can be executed
$TASKS$ = set of all tasks
To_r = set of all nodes to which resource r can travel
$TOC_{r,b}$ = cost of takeoff of resource r from base b, £
$Together_i$= set of tasks that must be completed together with index $i \in \{1..Ntog\}$
$TR0_r$ = initial value of the time slice at which resource r will run out of fuel
$TR_{r,j}$ = the time slice at which resource r will run out of fuel
Tsk_r = set of tasks that can be attended by resource r
$TT_{r,j,i}$ = number of time slices required by resource r to travel from node i to node j
$TIME$ = set of time slices
TW = time window size used in the simulation studies
$UPC0_r$ = fraction of the payload capacity that is used in resource r
$UPC_{r,j}$ = positive integer decision variable storing the fraction of payload capacity that is used in resource r at the time it arrives at node j
$VISIT$ = set of all visit tasks

$w_{j,t}$ = binary decision variable used to store the starting time for preemptive tasks
$xor_{j,k}$ = binary decision variable used to indicate if at least one of the tasks with XOR relationship has been performed
$x_{r,j,i}^{t,s}$ = primary binary decision variable
XOR_j = set of tasks that have XOR relationship between each other

I. Introduction

THE use of Unmanned Aerial Vehicles (UAV) for civilian applications is increasing rapidly, driven by both improvements in platform technology and the development in computational capabilities and communication technologies. Earlier work in UAV resource allocation has been largely focused on military operations, with comparatively less attention focused on civilian applications [1]. Platform autonomy continues to grow and systems are becoming gradually more reliable, however, commercial UAV operational models for applications such as disaster relief, commercial aerial photography, crop health monitoring and emergency response are inclined towards one user flying one UAV. The scope of current UAV civilian applications is driven by national regulatory requirements which typically restrict operations to small-size short-range vehicles within visual range of a safety pilot. Larger classes of UAV typically require more than one operator to perform a mission. A current research trend is the move from multiple operators managing one UAV to one operator managing multiple UAVs [1]. In the present work we develop a resource allocation system that offers multiple unmanned autonomous vehicle services to multiple users. This work is focused on multi heterogeneous UAVs complemented with ground and surface based autonomous vehicles.

The resource allocation system proposed in this article offers dynamically reconfigurable vehicle teams, which are temporarily formed to perform a mission or part of it. Teams get dismantled once the mission is serviced so that the team members can join various other teams attending different missions. The majority of current UAV resource allocation systems tend to be problem specific, focusing on, for example, wide-area search, sense and react, or data gathering missions. Each of these missions has a set of UAVs associated with it that can only perform that particular mission. The use of a problem specific approach is a domain bound exercise that simplifies the problem at the expense of universality. Problem specific planning solutions generally do not accommodate the operational constraints imposed by multiple mission scenarios. For instance Murray et al. [2] considered a fuel quantity constraint and constraints on time windows for tasks, however, refuel scheduling and sensor selection options are not included. Mufalli et al. [3] considered sensor selection, finite fuel resources and prescribed time windows, however refuel scheduling, dynamic sensor selection, heterogeneous task types and cooperation between UAVs are not included. A multi-user system requires the ability to accommodate multiple mission types that may be requested in parallel. During the planning phase, the proposed system must be able to generate an efficient action plan for each available resource, a refueling schedule and a sensor selection schedule that permits change of sensors during execution. System operators have different perspective on efficiency, while some may prefer reducing cost other may prefer reducing mission time.

The resource allocation system introduced in this paper incorporates the ability to plan point to point transportation missions with a single transshipment node in conjunction with sensing and data gathering missions. The efficiency of the solution is controllable by the operator through varying the importance of time versus operational cost to suit the overall goals of the operation. The resource allocation system has the ability to replace active resources with more effective ones once they become available. It also enables automation of the interaction between the user and the vehicle fleet and offers an automated asset management solution for the system operator.

We recognize that whilst there are benefits in producing a generic resource allocation solution, there are some instances where pre-configured solutions may be more appropriate. For example surveillance problems requiring search and tracking of moving targets may benefit from fast distributed systems such as those presented in [4][5][6]. Plume source problems can benefit from multi UAV coordination systems based on swarm intelligence similar to the one presented in [7]. Specialized transportation missions requiring more than one transshipment node or aiming to cover very large area can use network models similar to the one presented in [8]. Other applications that use UAVs for cooperative slung load transportation may benefit more from solutions such as the distributed system developed as part of the AWARE project [9].

It is understood that the emphasis of the proposed resource allocation system is on production of an efficient plan without particular regard to the computational cost of developing the plan. Furthermore, in order to enable dynamic resource allocation, it is necessary that the computing resource undertaking the planning has full and continuous access to bidirectional communication with all the vehicle assets involved.

Section II provides reviews of multi-UAV systems and use of ILP for resource allocation. The problem definition and description of the proposed allocation system is presented in section III. The ILP formulation of the resource allocation problem is presented in section IV. Section V presents simulation results and provides a discussion of the performance and capabilities of the developed system. Finally, the main outcomes of the article are summarized in the conclusion in section VI.

II. Literature Review

Research into operations models for multi UAV systems in military scenarios has been active since the 1970's as described by Aume et al. [10]. The reader is referred to Maza et al. [11] who presents some main advantages of using multiple UAVs in comparison to a single powerful one, with a good classification of multi-UAV systems based on the coupling between UAVs. Design consideration for multi-UAV planning architectures are covered in details by Ponda et al. in [12] with a review on Linear Programming (LP), Markov Decision Process (MDP) and Game Theory approaches presented in the context of solving cooperative mission planning for multi-UAV teams.

In civilian operations, we find many multi-UAV applications typically focused on acquisition, processing and distribution of sensor data. Applications include surveillance and patrolling [13] [14], live image capture [15], environment monitoring [16], area exploration [17] [18] and radiation mapping [19]. Dynamism in sensing applications comes from new requests and task modifications issued by the end user or from fault conditions in any of the UAVs [17].

Dynamic resource allocation systems have the ability to reallocate tasks and resources during mission execution. The ability to reallocate tasks can lead to the potential problem of repeated reassignment known as 'churning'. This decision cycling wastes time and may lead to infeasible solutions as shown in [20]. Different strategies have been developed to address the problem of churning such as use of a filter that limits the rate of change of the task assignment [20] or use of an additional weighing that favors the current plan over new plan as described in [1].

All linear programming approaches, whether centralized or not, may suffer from the curse of dimensionality. In our case, the number of decision variables grows linearly with the number of vehicles and the number of time slices keeping the problem theoretically tractable. However, processing time typically increases exponentially with the increasing number of UAVs and tasks [21]. Nonetheless, the use of a ground based central computing point means that the cost of computational power is reduced. Whilst some studies consider a centralized system to be a single point of failure causing reduced robustness as in [22] [23], this is mitigated by the fact that a system on the ground affords high equipment quality with redundancy in hardware, servers and operators [12]. Cummings [22] claims that decentralized systems are much harder to certify as safe compared to centralized ones, making a centralized system more appealing for use in a civilian airspace. Decentralized systems are preferred for missions with communication constraints, e.g. [24] [25] [26] . Decentralization is also beneficial when there is little or no input required from the ground operator [7]. Nevertheless, if the operator is an integral part of the mission, there is always a need to send important information back to ground. The coupling presented in some scenarios, especially time coupling and assignment-path coupling render decentralized systems very complex to design, hence making the centralized system more suitable in such cases [2] [27]. Communicating UAV situational awareness has been the motivation for many algorithms such as [28] [29] due to its critical importance in the military battlespace. For civil applications, sharing the situational awareness is important but less critical.

For the type of problems this paper is addressing, linear programming offers a convenient mathematical modeling method that can express the resource allocation problem as an optimization problem with linear constraints [12]. UAV LP formulations such as [30][31] attempted to solve task allocation and trajectory planning as a coupled problem. Current practice tends to decouple the problem, focusing primarily on task allocation. Trajectory planning is then undertaken based on simple Euclidean paths (in most cases) [32]. This allows the incorporation of specialized path planning algorithms into the planning system such as ground tracking paths [33], obstacle avoidance paths [34] and paths for area coverage [18], which may be more important than trying to find integrated solutions.

The work presented by Shima et al. [1] shows use of MILP models to solve problems which are quite dynamic in nature. To account for this dynamism, solution methods need to respond in real-time and therefore speed prioritized solution methods such as a Genetic Algorithm solver and Tree Search approach were used. However, the formulations in [1] only considered a limited number of available UAVs (with their associated payload capacity and dynamic constraints), while refueling and other operational constraints were not considered. The study by Faied et al. [35] formulates the question of multi-UAV task allocation as an instance of the classical Vehicle Routing

Problem (VRP). A Receding Horizon approach is used by Bethke et al. [36] to solve an ILP sequence over the period of its execution (planning horizon) and also executes them over a shorter action horizon, allowing the planner to incorporate refueling tasks into the operation. This approach was also used by Nigam et al. [37] to perform persistent surveillance operations and by Kim et al. [14] to manage UAV visits to service stations while performing persistent monitoring missions. The work in [14] uses a branch and bound technique to solve the MILP formulation in real-time. This work was extended in [38] to allow the UAVs to start from arbitrary initial conditions, and a new solution approach called Sequential Task Assignment Heuristic (STAH) is proposed in order to obtain faster response that is near optimal.

Another approach, by Mufalli et al. [3], focuses on selecting the most appropriate sensor combination to have on UAVs prior to the mission execution. Its main emphasis is on efficient planning of surveillance missions. The problem presented in [3] was solved using different types of column generation heuristics in order to rapidly obtain a near optimal solution.

The framework presented by Murray et al. [2] provides a universal model for military operations. Although the authors considered fuel capacity constraints, they did not consider refueling the UAVs on low fuel allowing the UAVs in special cases to crash on low fuel as long as the mission succeeds. Murray et al. later developed a branch and bound solver for his model [39] which reduces the processing time and allow for real time responses.

The transportation problem for UAVs is not tackled in the literature at the same depth as the sensing applications have been for military missions. This is in part due to the high resemblance to current ground based transportation problems which have existing solutions; many of the available VRP solutions apply to UAVs in the presence of an appropriate path planner and path estimator[40]. A great deal of work has been done in cooperative sling load transportation such as the work of Michael et al. in [41] or the work of Maza et al. in [42]. The major challenge in this type of application is the routing problem in the presence of different vehicle capabilities. Different logistics paradigms can be implemented based on representing the problem as VRP with Pickup and Delivery (VRPPD) [35], multi depot Traveling Salesman Problem (mTSP)[43], Capacitated Pickup and Delivery Problem [44] and many more. Alternatively, other methods exist such as a novel approach inspired by communication network routing solutions recently proposed by Raptopoulos [8]. The problem of cooperating on a heavy load poses many challenges in multi UAV systems which are beginning to be approached [9]. The system proposed in this paper learns from VRPPD [35] and the work in [8] in order to create a complete solution capable of dealing with sensing and transportation mission types together.

Finally, many algorithms are addressing a specific aspect of the problem like time dependency [31], communication constraints and failure [25][13], heterogeneous capabilities [29], distribution [45] and scalability [46]. In the present work, we use a combination of different approaches presented in various references [3] [2][43] [47] in order to answer the problems posed earlier. The framework presented by Murray et al in [2] provides a very good starting point for this formulation since it includes all the basic notions and definitions that we are proposing. Although the modifications we add to the framework in [2] are substantial, it can still be referred to for understanding of concepts. The sensor selection problem has been addressed in isolation in [3] which was only focusing on ISR operation within the endurance of the UAVs. As for the refueling constraints, a truck delivery system have been considered and described in [47], however the formulation does not include any of the problems faced by allocating resources for multiple heterogeneous UAVs.

III. System Overview

In this section, we describe the main requirements of the resource allocation system and how it relates to other components of a civilian heterogeneous vehicle system with multiple users. In order to ensure safety and control over the vehicle fleet, there must be a ground control center that has continuous ability to monitor and control all vehicles at all times. This requires information about all UAVs to be channeled to the ground to provide full situational awareness of the UAV fleet. Figure 1 presents a proposed architecture where Global and Local decision making systems are fused into a hybrid Resources Management System (RMS) for a fleet of unmanned {x:Aerial, Ground or Surface} vehicles. The centralized global decision making system is used to generate efficient schedules for all resources while the local distributed decision making system takes over the coordination decisions that are crucial for the success of the mission. The "Operation Manager" module allows identification of vehicle health and associated mission risks. The vehicle fleet information can be combined with information from air traffic controllers to ensure a safe and smooth operation between airspaces.

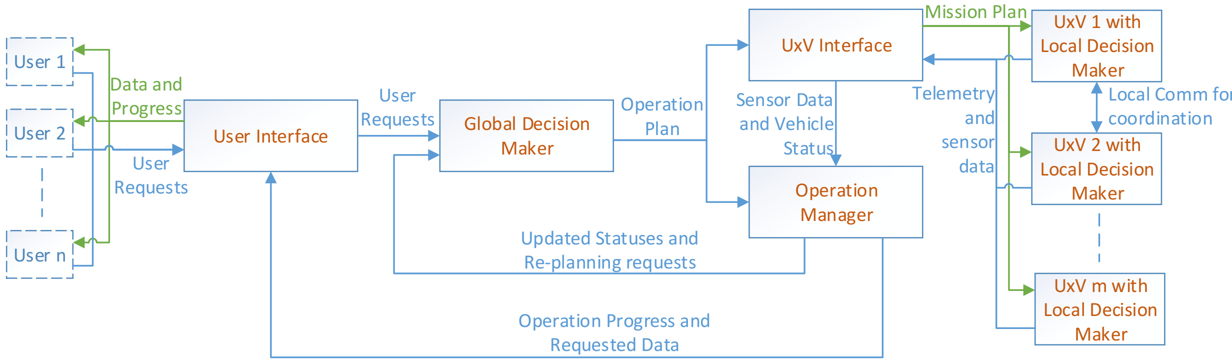

Fig. 1: Resource management system architecture.

The “Global Decision Maker” module consists of three different interconnected sub-modules shown in Fig. 2: Pre-planner, Resource Allocator and Trajectory Planner. The pre-planner takes user requests and provides elemental tasks and traveling time estimates to the resource allocator. The trajectory planner is responsible for planning a collision free efficient path for each vehicle in order to fulfill the mission requirements and follow the schedule planned by the resource allocator. The focus of this article is on the design of the Resource Allocator sub-module. The presented resource allocation model incorporates the most important constraints that need to be considered for planning an effective, efficient and robust multi-vehicle operation serving multiple users simultaneously.

The resource allocator is required to map on the following principal performance attributes:

- Effectiveness: The solution should meet the user mission outcome requirements (independent of cost) by allocating resources that maximize the quality of service. To be effective the resource allocator must be able to handle heterogeneity in vehicles and sensors, and must be dynamic in that it is able to accommodate modification and removal of user requests and re-plan in case of vehicle faults.
- Efficiency: The solution cost is minimized based on the mission cost model defined by the operator.
- Robustness: Solution plans remain feasible under real world variations in the mission model.

The resource allocator must account for vehicle fuel capacity, ensuring vehicles are refueled at optimal points in time and ensuring all UAVs safely land before running out of fuel. It must accommodate multiple take-off and landing sites with availability limitations. The resource allocator must also permit system operations beyond individual vehicle endurance limits by ensuring smooth transition between depleted resources and fresh ones.

We choose to use a task-focused model rather than a vehicle focused model to allow a task to be served by any capable set of available vehicles. The problem under consideration incorporates the presence of multiple users attempting to use the system in real time, together with the optimization of multiple competing objective functions imposed by the business implication of the vehicle sharing system and the operational requirements of the system.

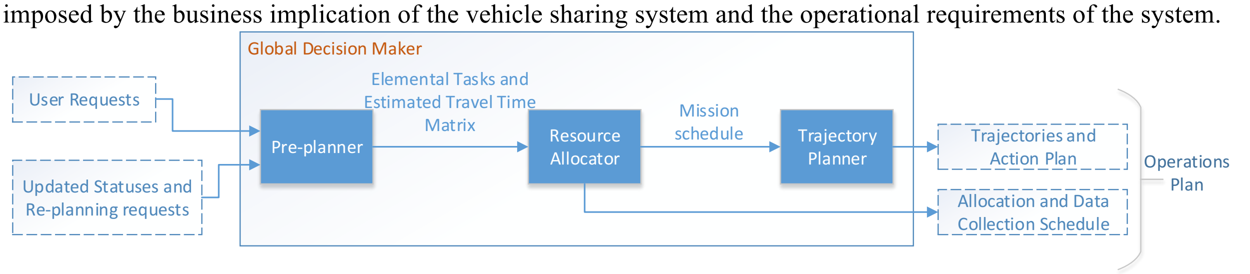

Fig. 2: Internal architecture of the Global Decision Maker block.

A user request is a composition of single or multiple tasks, with the assumption that requests are independent from each other. Each request is decomposed into simple elemental tasks that constitute the basis of the input to the resource allocator. Request tasks are processed by the pre-planner sub-module to exploit self-evident synergies such as grouping geographically adjacent sensing activities into a single sensing activity. In this environment, vehicle resources are shared amongst unrelated requests while ensuring the success of each.

The pre-planner sub-module calculates travelling time between all possible tasks, taking into account the starting and finishing point of each task. The pre-planner can employ any implementation of a quick path planning technique capable of estimating time required for a vehicle to travel between two points. All possible sensor combinations are evaluated taking into account their effectiveness over each task, and their suitability for each platform. The pre-planner calculates remaining endurance of each vehicle, take-off and landing energy consumptions and endurance reduction due to extra payload weight. The pre-planner design and suitable task processing algorithms are not covered in this article.

The resource allocator answers the questions of "Which vehicle resources are needed to perform each of the tasks? When should those resources attend each task? When should the vehicles go to their bases?" The types of tasks included in our formulation are *Visit*, *Loiter*, *Transport*, *Intersection* and *Preceded* tasks. *Visit* tasks are tasks requiring a resource to perform a single operation starting at a certain waypoint and ending at another, e.g. scanning a section of a river border. In cases where a single instantaneous operation is required such as taking an image, the start and end points for *Visit* are the same. *Visit* tasks that are constrained by the completion of other tasks are referred to as *Preceded* tasks. *Loiter* tasks require a resource to perform an action continuously over a certain specified site. The site can be defined as a geometric area with an entry and exit waypoints. This is convenient for applications such as persistent surveillance. Alternatively, the site can be defined as a single point of interest through specifying a central waypoint and a loiter radius around it. When the physical solution space of two tasks intersects, there exists a geographical location that can be visited once in order to complete both tasks in a single operation. Such tasks are defined as *Intersection* tasks. Apart from sensing tasks, we also include *Transport* tasks that offer a point to point delivery service with the ability to use a base as a transshipment node. Finally, *Visit, Transport*, *Intersection* and *Preceded* tasks are non-preemptive tasks hence they cannot be paused, and must be completed in full using the resource they are first allocated to. *Loiter* tasks are preemptive tasks which allow exchange of resources as long as at least one resource is allocated to the task during the execution of the task.

In the formulation in section IV, a base offers a number of services to the vehicles. It allows them to refuel, exchange cargo and change sensors. The formulation allows the allocation system to plan for sensor change at bases depending on the needs of the mission; sensors can be installed as a suite or individually. The sensor selection decision takes into account the weight of the sensors and the subsequent effect on the vehicle endurance, the capacity and capability of the UAV to carry the sensors, the cost of installing and operating them and the compatibility between sensors installed together.

The presented task definitions reduce the number of tasks needed to describe user requests and therefore reduce the size of the problems considered; this reduction makes it possible to include more constraints for the same solution time particularly refueling at bases and payload selection constraints. This permits the Global Decision Maker module to plan for a period of time beyond individual vehicle endurance without being limited by the type of sensors on-board. The bases are considered to have the ability to accommodate as many vehicles as required, however the structure of the formulation allows for managing congestion in the future by varying base availability time windows in which vehicles are allowed to land, refuel and be maintained.

IV. Model Formulation

In keeping with previous work on Vehicle Routing Problems, the resource allocation problem is modeled as a connected network of nodes in which each node represents a position and a characteristic related to that position whether it is a task, base or initial position. Figure 3 presents an example of a network formed by two bases, two non-base initial positions, four tasks and four resources. The resources travel amongst the connected nodes receiving benefits from tasks achieved at certain nodes and expending cost due to travelling, maintenance and sensor operation. Nodes have different requirements and constraints, and will affect the resources in different ways.

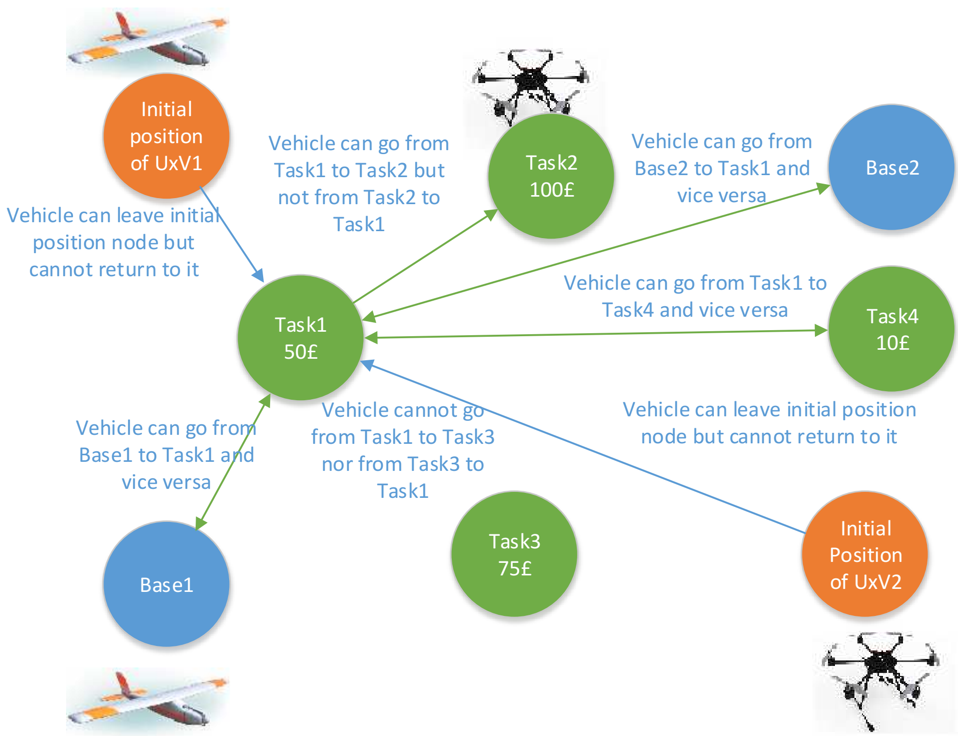

Fig. 3: Example of a connected network of nodes with 4 vehicles positioned at different nodes.

For the present model we use a discretized time approach which divides time into equal time slice durations. This offers the planner the ability to change the planning time resolution by varying the real world duration of the time slices. High resolution plans will have a very large number of variables to calculate and will only be feasible to compute for short time horizons. Alternatively, an indicative plan for longer time horizon can be obtained using coarser time slices. The effect of varying the time resolution is discussed in section V.

A. Time Definition

The number of time slices used for planning is defined in the variable NTS which is used to generate an ordered integer set of time slices $\{0 \ldots NTS\}$ defined as the set $TIME$.

B. Node Definition

Nodes model the following three categories: Bases, Initial Positions and Tasks.

1. Bases

Nodes in the set $BASES$ can accept vehicles for servicing, refueling and cargo transshipment. This provides an extension to the type of bases presented in [2]. Base nodes are only available to accept vehicles during specified periods of time represented by $T_b \subseteq TIME$.

2. Initial Positions

We define the nodes for the initial position of each resource r in the set $InitPos_r$. $InitPos_r$ may contain bases nodes if the resource is starting from a base. If the resource is not starting from a base, a new node is defined representing the current position of the resource as its initial position.

3. Tasks

We define $TASKS$ as the set of all requested tasks. Nodes represented in $TASKS$ can be part of one of the following subsets.

a. <u>Visit</u>

The basic most versatile task type is defined in the set $VISIT$. Two subtypes are considered:

i. A single almost instantaneous operation such as taking a single reading or a snapshot at a certain waypoint within a window of time specified in $T_j \subseteq TIME$ and $j \in VISIT$.

ii. A one-time acquisition operation along a predefined path set by a starting and ending waypoint that must start within a window of time specified in T_j.

A non-preemtable task causes a vehicle resource to be exclusively engaged on a task until the task is completed. The visit task has a duration of execution l_j ($j \in VISIT$) that accommodates both subtypes using the same format: In (i) l_j is set to one time slice, while in (ii) it is set to the estimated number of time slices required to follow the required path.

b. Loiter

Loiter tasks are defined in the $LOITER$ set. *Loiter* tasks require a vehicle resource to perform a continuous action at a given loiter radius at a given waypoint. The duration of execution l_j with $j \in LOITER$ specifies the number of time slices the operation should last for. This type of task is preemptable to allow other vehicles to take over the task where the duration of the current vehicle is exhausted, or to swap with a capable vehicle so that the current vehicle performs a higher priority task.

c. Transport

Transport tasks are used to deliver a cargo to a delivery point defined as a task in $DELIVER$ after it has been picked up from a point defined as a task in the set $Pick_j \subseteq VISIT \cup BASES$ with $j \in DELIVER$. The cargo weight is specified in PW_j. When reallocating during execution, $Pick_j$contains the initial position $InitPos_r$ of the resource r that already picked up the cargo.

d. Intersection

The pre-planner processes all user requests to find any correlations between sensing activities. Geographically grouped sensing activities can be combined into a single sub-mission hence creating a single task that is sufficient to attend instead of attending the grouped tasks individually. For example, receiving two requests from two different users for monitoring two very close points within the same period of time can be combined into one request that serves both users. The resource allocator has the option to either perform each request individually or perform the combined request that serves both users together. The resource allocator cannot choose to perform the combined request and one of the individual requests. A special type of tasks is defined to describe this relationship.

$VISIT$ tasks that can potentially have an *Intersection* task are defined in $PotINTER$. The *Intersection* tasks are those tasks that if attended obviate the need to attend two or more of the $PotINTER$ tasks. *Intersection* tasks are defined in the set $INTER$. Furthermore, the resource allocator must decide between doing either the original task in $PotINTER$ or the corresponding *Intersection* task in $INTER$, therefore tasks in these two sets have a xor relationship.

Define XOR_j as the set of tasks that have XOR relationship between each other with $j \in PotINTER$ and $XOR_j \subseteq INTER \cup PotINTER$. In order to perform task j the resource allocator has to choose only one of the tasks in XOR_j. For example, take task A $\in PotINTER$ and task B $\in PotINTER$ and define the task combining task A and task B to be the *Intersection* task denoted task A $\cap$ B $\in INTER$. The xor relationships between the original tasks and their intersection task are then captured in the sets $XOR_A = \{A, A \cap B\}$ and $XOR_B = \{B, A \cap B\}$

e. Preceded

Some tasks may have dependencies on preceding tasks hence we define $PRECEDED$ as the set of tasks that cannot start execution before at least one preceding task has been completed. $PRECEDING_j$ is the set containing the tasks that precede task $j \in PRECEDED$ with $PRECEDING_j \subseteq VISIT \cup LOITER \cup DELIVER \cup PotINTER \cup INTER \cup PRECEDED$ while j is treated as a $VISIT$ task. A lag time can be set from the time the preceding task is completed to the time the preceded task starts through lag_j.

We therefore have the set $TASKS = VISIT \cup LOITER \cup DELIVER \cup PotINTER \cup INTER \cup PRECEDED$. We can state that for all nodes $j \in TASKS \cup BASES$, $T_j \subseteq TIME$ is the set of time slices in which j can be attended. Moreover for all $j \in TASKS$, G_j is the gain or benefit of performing task j and l_j is the number of time slices required to complete task j. Furthermore, we define the set of non-preemptive nodes $NPre = VISIT \cup BASES \cup DELIVER \cup PotINTER \cup INTER \cup PRECEDED$ and the set of preemptive nodes $Pre = LOITER$.

Finally, in order to explicitly enforce heterogeneous resources cooperation we define the set $Together_i$ being a subset of $NPre$ and containing tasks that must be completed together hence $Together_i \subseteq NPre$ and $i \in \{1..Ntog\}$ with $Ntog$ being the number of task sets that need to be completed together.

C. Resources

Vehicles are grouped in the set $RESOURCES$. Resources are allowed to carry different types of sensors and transport cargo. They have different platform configurations such as fixed wing, rotary wing or ground vehicle allowing each to do certain jobs with better efficacy. We define R_j as the set of resources that have the capability to attend node $j \in TASKS \cup BASES$ with $R_j \subseteq RESOURCES$, and define B_r the set of bases that can be visited by resource $r \in RESOURCES$ with $B_r \subseteq BASES$. We also define $Tsk_r \subseteq TASKS$ to be the set tasks that can be attended by resource r. Using these definitions we define $To_r = B_r \cup Tsk_r$ as the set of nodes to which resource r can travel. Finally, the nodes from which resource r can travel to node $j \in To_r$ are grouped in $From_{r,j} = if\ j \in Pre\ then\ (InitPos_r \cup B_r \cup Tsk_r)\ else\ ((InitPos_r \cup B_r \cup Tsk_r) - \{j\})$.

Each resource has a limited amount of fuel encoded as the available duration in number of time slices. Fuel can be replenished whilst the vehicle is at a base. The maximum fuel capacity of each resource is Q_r and $Q = \max(Q_r)$ while the payload capacity is CAP_r. Each resource also has a platform effectiveness value taken between 0 and 1 defined in $PE_{r,j}$ for $r \in RESOURCES$ performing task $j \in TASKS$. If a platform is not suited for a certain task, $PE_{r,j}$ will be set to -1. The value of $PE_{r,j}$ for unsuitable resources has no effect of the ILP model since only capable resources are considered for allocation, however the -1 value helps identify capable resources in the preplanning phase when R_j and Tsk_r are generated. The effectiveness of a platform for a given task affects the value of the benefit acquired from that resource attending that task.

The effectiveness of a resource is the measure of how good this particular platform is at performing a particular task. For example, monitoring a road junction can be done more effectively using a multi rotor. Even though a fixed wing aircraft can perform the monitoring task, the changing orientation of the view point is a disadvantage. This may or may not be acceptable and therefore the effectiveness measure is there to capture this preference on a continuous scale between 0 and 1.

Vehicle operation incurs costs that are quantified in a way to represent real costs. All the costs are encoded as depreciation costs proportional to the operation time, and are set using the value of maintenance cost per time slice for operating resource r, MC_r. Take-off and landing may account for a large portion of the vehicle operating cost, which may include cost of operating from a given base, the ground crew cost for the operation, and various fees. For resource $r \in RESOURCES$ from base $b \in B_r$ define $TOC_{r,b}$ and $LC_{r,b}$ as the cost of take-off and landing of resource r from/at base b respectively. Any additional operational cost that is platform or base specific is included in the additional operational cost, $AOC_{r,b}$.

Time taken to travel between nodes is the main parameter the resource allocator uses for scheduling. The estimated travel time computed by the pre-planner is then taken by the resource allocator as a matrix of time slices $TT_{r,j,i}$ containing the number of time slices required by $r \in RESOURCES$ to travel to node $j \in To_r$ from node $i \in From_{r,j}$. To simplify the formulation, take-off and landing times are included in $TT_{r,j,i}$ so that allocation need only considers airborne time. Additionally, in the case of $NPre$ tasks, the requested execution time l_j is added to $TT_{r,j,i}$ for all possible $j \in To_r$. In this case, the resource allocator assumes an execution time of 1 time slice.

Starting conditions are defined in $TR0_r$, the number of time slices available for allocation based on the remaining fuel in the resource, and $UPC0_r$ representing the fraction of the payload capacity that is used in resource $r \in RESOURCES$ at the beginning of the planning. $UPC0_r$ must be carefully calculated in case r has picked up a cargo in previous plans since failure to use the correct weight may lead to incorrect allocation.

D. Sensors

The overall capability of a vehicle resource is dependent on the vehicle capability combined with the capability of the sensors on-board. The set $SENSORS$ contains all sensor combinations that are possible considering the available sensors, platform payload limitations and constraints regarding which sensors can be installed together on the same platform. An empty payload bay is taken as a type of sensor combination to reduce formulation complexity. Each individual resource r has a limited set of sensor combination S_r that can be installed on it. The initial sensor combination installed on-board resource r at the beginning of the planning is $InitSen_r \subseteq S_r$.

The weight and power consumption of each sensor will affect the platform endurance, therefore we use $SR_{r,s}$ to represent the reduction in number of time slices available to a platform due to the installation of sensor $s \in S_r$

onboard resource $r \in RESOURCES$. The cost of operating a sensor is captured by the sensor usage cost SUC_s which represents the cost per time slice of operating sensor $s \in SENSORS$.

Finally, some sensors are more suitable for certain tasks than others hence we include a sensor effectiveness parameter $SE_{s,j}$ representing the effectiveness of sensor s in performing task $j \in TASKS$. This value is coupled with $PE_{r,j}$ in order to denote an overall resource effectiveness when performing a particular task. When attending task j, $SE_{s,j}$ takes values between 0 and 1 for appropriate sensors or a large negative number if the sensor is not fit for the task.

E. Variables

Define $x_{r,j,i}^{t,s}$ to be the primary binary decision variable of the proposed model. A value of 1 indicates that a resource $r \in RESOURCES$ is assigned to travel from node $i \in From_{r,j}$ to node $j \in To_r$ servicing node j at time $t \in T_j$ equipped with sensor combination $s \in S_r$. Hence we can write

$$x_{r,j,i}^{t,s} \in \{0,1\} \ \forall\, r \in RESOURCES,\ j \in To_r,\ i \in From_{r,j},\ t \in T_j,\ s \in S_r \tag{1}$$

Define $TR_{r,j}$ to be a positive integer representing the time slice at which fuel will run out from resource $r \in RESOURCES$ measured at the time it arrives at node $j \in (To_r \cup InitPos_r) - BASES$, i.e. the last time which the resource can be allocated to tasks. This value is carried on from one node to the other until a base is reached at which point a new value is calculated and carried forwards. Note that the time taken to land and take-off during a fueling stop are accounted for in the value of $TR_{r,j}$.

$$TR_{r,j} \geq 0 \ \forall\, r \in RESOURCES,\ j \in (To_r \cup InitPos_r) - BASES \tag{2}$$

Define $w_{j,t}$ as a binary decision variable used to store the starting time for preemptive tasks. This variable is used to ensure loiter tasks are performed in consecutive time steps. A value of 1 indicates that task, i.e. node $j \in Pre$ is visited at time $t \in T_j$.

$$w_{j,t} \in \{0,1\} \ \forall\, j \in Pre, \qquad t \in T_j \tag{3}$$

Define d_j as a binary decision variable, used to indicate if the payload to be delivered at $j \in DELIVER$ has been picked up from $Pick_j$ (delivery precedence constraint).

$$d_j \in \{0,1\} \ \forall\, j \in DELIVER \tag{4}$$

Define $dt_{j,t}$as a binary decision variable used to store the pickup time of the payload to be delivered at $j \in DELIVER$ (delivery time precedence constraint). A value of 1 indicate that task $i \in Pick_j$ was completed at time $t \in T_j$.

$$dt_{j,t} \in \{0,1\} \ \forall\, j \in DELIVER, \qquad t \in T_j \tag{5}$$

Define $UPC_{r,j}$ as a positive integer decision variable storing the fraction of payload capacity that is used in resource $r \in RESOURCES$ measured at the time it arrives at node $j \in (To_r \cup InitPos_r) - BASES$. This value is carried on from one node to the other until a base is reached or a delivery point is attained whereupon a new value is calculated and carried forwards.

$$UPC_{r,j} \in \mathbb{Z}^+ \ \forall\, r \in RESOURCES, \qquad j \in (To_r \cup InitPos_r) - BASES \tag{6}$$

Define $xor_{j,k}$ as a binary decision variable used to indicate if at least one of the tasks with XOR relationship has been performed. A value of 1 indicate that for task $j \in PotINTER$ task $k \in XOR_j$ has been allocated hence j doesn't need to be allocated.

$$xor_{j,k} \in \{0,1\} \ \forall\, j \in PotINTER, \qquad k \in XOR_j \tag{7}$$

Define px_j as a binary decision variable used to indicate if any preceding task of $j \in PRECEDED$ has been allocated.

$$px_j \in \{0,1\} \ \forall\, j \in PRECEDED \tag{8}$$

Define $pxt_{j,t}$ as a binary decision variable used to store the time $t \in T_j$ at which the preceding task of $j \in PRECEDED$ has been allocated.

$$pxt_{j,t} \in \{0,1\} \ \forall\, j \in PRECEDED, \qquad t \in T_j \tag{9}$$

Define $Fintime_r$ as an integer variable representing the finishing time slice of each resource $r \in RESOURCES$. This is used to reduce waiting and traveling time of individual resources.

$$0 \le Fintime_r \le EndTime \ \ \forall\, r \in RESOURCES \tag{10}$$

Define $Comptime$ to be a positive integer variable representing the overall completion time of the mission.

$$Comptime \ge 0 \tag{11}$$

F. Task Constraints

1. Visit

$$\sum_{r \in R_j} \sum_{t \in T_j} \sum_{i \in From_{r,j}} \sum_{s \in S_r} x_{r,j,i}^{t,s} = 1 \ \ \forall\, j \in VISIT \tag{12}$$

Equation (12) ensures that each visit task must be attended once by a capable resource.

2. Loiter

Equations (13) and (14) state that a *Loiter* task must be attended by no more than two resources and no less than one resource at any time slice during the execution of the task. This allows resources to change over when servicing a *Loiter* task while ensuring there is at least one resource attending the task at any time slice. Thus, loitering beyond the endurance of a single resource becomes possible.

$$\sum_{r \in R_j} \sum_{i \in From_{r,j}} \sum_{s \in S_r} x_{r,j,i}^{t,s} \le 2 w_{j,t} \ \ \forall\, j \in LOITER\,, \qquad t \in T_j \tag{13}$$

$$\sum_{r \in R_j} \sum_{i \in From_{r,j}} \sum_{s \in S_r} x_{r,j,i}^{t,s} \ge 1 w_{j,t} \ \ \forall\, j \in LOITER\,, \qquad t \in T_j \tag{14}$$

Equations (15) and (16) make sure a *Loiter* task can only be visited l_j consecutive times in total by all allocated resources. This ensures all the resources cooperating on the *Loiter* task will be allocated in one continuous time window.

$$\sum_{t \in T_j} w_{j,t} = l_j \ \forall\, j \in LOITER \tag{15}$$

$$\sum_{ta \in T_j : ta \notin [t - l_j + 1, t + l_j - 1]} w_{j,ta} \le \left(1 - w_{j,t}\right) l_j \ \ \forall\, j \in LOITER, \qquad t \in T_j \tag{16}$$

3. Transport

Inspired by [35], [48] and [47], the following constraints for the transportation tasks were developed. Equation (17) sets d_j to 1 when the cargo pickup task is completed, then Eq. (18) allows the cargo to be delivered once it has been picked up. The delivery task must be completed if and only if the pickup task is completed. If the cargo has already been delivered to a transshipment base, Eq. (19) makes sure the delivery is initiated from this base. On the other hand, if the cargo is already with resource r, Eq. (20) allows the delivery to take place knowing that the cargo is at $InitPos_r$ at the beginning of the planning. Eq. (21) and (22) ensure the pickup happens at least 1 time slice before the allocated delivery time. Since the delivery must be done by the same resource as pick up, traveling time from pick up to delivery point is automatically added.

$$\sum_{k \in Pick_j \cap VISIT} \sum_{r \in R_k} \sum_{i \in From_{r,k}} \sum_{s \in S_r} \sum_{t \in T_k} x_{r,k,i}^{t,s} = d_j \quad \forall j \in DELIVER: Pick_j \subseteq VISIT \tag{17}$$

$$\sum_{r \in R_j} \sum_{i \in From_{r,j}} \sum_{s \in S_r} \sum_{t \in T_j} x_{r,j,i}^{t,s} = d_j \quad \forall j \in DELIVER : Pick_j \subseteq VISIT \tag{18}$$

$$\sum_{r \in R_j} \sum_{i \in (B_r \cap Pick_j)} \sum_{s \in S_r} \sum_{t \in T_j} x_{r,j,i}^{t,s} = 1 \quad \forall j \in DELIVER : Pick_j \subseteq BASES \tag{19}$$

$$\sum_{r \in R_j} \sum_{i \in (InitPos_r \cap Pick_j) \cup BASES} \sum_{s \in S_r} \sum_{t \in T_j} x_{r,j,i}^{t,s} = 1 \quad \forall j \in DELIVER : Pick_j \nsubseteq VISIT \cup BASES \tag{20}$$

$$\sum_{r \in R_j} \sum_{i \in From_{r,j}} \sum_{s \in S_r} x_{r,j,i}^{t,s} = dt_{j,t} \quad \forall j \in DELIVER, t \in T_j : Pick_j \subseteq VISIT \tag{21}$$

$$\sum_{r \in R_k} \sum_{i \in From_{r,k}} \sum_{s \in S_r} \sum_{tk \in T_k : tk > t-1} x_{r,k,i}^{tk,s} \leq \left(1 - dt_{j,t}\right) \forall j \in DELIVER, k \in Pick_j \cap VISIT, t \in T_j : Pick_j \subseteq VISIT \tag{22}$$

The presented model allows one transshipment at a Base node, therefore we introduce Eq. (23) to ensure that cargo has been picked up before a transshipment at the base and Eq. (24) to ensure that the resource carrying the cargo in a previous plan is the one transshipping at the base. Finally, Eq. (25) allows delivery from either the pickup point or the transshipment base.

$$\sum_{i \in Pick_k \cap VISIT} \sum_{ra \in R_i : j \in B_{ra}} \sum_{tj \in T_j : tj + TT_{r,k,j} \leq tk} \sum_{s \in S_r} x_{ra,j,i}^{tj,s} \geq \sum_{s \in S_r} x_{r,k,j}^{tk,s} \ \forall k \in DELIVER, r \in R_k, j \in B_r, tk \in T_k : Pick_j \subseteq VISIT \tag{23}$$

$$\sum_{i \in Pick_k - VISIT} \sum_{ra \in RESOURCES : j \in B_{ra}, InitPos_r \subseteq Pick_j} \sum_{tj \in T_j : tj + TT_{r,k,j} \leq tk} \sum_{s \in InitSen_r} x_{ra,j,i}^{tj,s} \geq \sum_{s \in S_r} x_{r,k,j}^{tk,s} \ \forall k \in DELIVER, r \in R_k, j \in B_r, tk \in T_k : Pick_j \nsubseteq VISIT \cup BASES \tag{24}$$

$$\sum_{r \in R_k} \sum_{i \in From_{r,k} - (Pick_k \cup BASES)} \sum_{s \in S_r} \sum_{t \in T_k} x_{r,k,i}^{t,s} = 0 \quad \forall k \in DELIVER \tag{25}$$

Each resource has a payload capacity that must not be exceeded. When a cargo is picked up, the capacity is reduced, then when the cargo is delivered to a base or a delivery point, the capacity is increased. Eq. (26) updates the used capacity of each resource as soon as it attains a node j.

$$UPC_{r,i} + \left(x_{r,j,i}^{t,s} \times PW_k\right) \leq UPC_{r,j} \ \forall k \in DELIVER, j \in Pick_k \cap VISIT, r \in R_j, i \in From_{r,j} - BASES : i \neq j, \quad t \in T_j, \quad s \in S_r \tag{26}$$

Equation (27) and (28) consider a transportation resource picking up a cargo from a base. Eq. (27) resets the fraction of the payload capacity used to the weight of the cargo being transported while Eq. (28) enforces the resource capacity limit on the cargo carried.

$$\left(x_{r,j,i}^{t,s} \times PW_k\right) \leq UPC_{r,j} \; \forall\, k \in DELIVER, \quad j \in Pick_k \cap VISIT, \quad r \in R_j, \quad i \in From_{r,j} \cap BASES : i \neq j, \quad t \in T_j, \quad s \in S_r \tag{27}$$

$$\left(x_{r,j,i}^{t,s} \times PW_j\right) \leq CAP_r - UPC_{r,j} \;\; \forall\, j \in DELIVER, \quad r \in R_j, \quad i \in From_{r,j} \cap BASES : i \neq j, \quad t \in T_j, \quad s \in S_r \tag{28}$$

Equation (29) reduces the used capacity once the cargo has been delivered.

$$UPC_{r,i} - \left(x_{r,j,i}^{t,s} \times PW_j\right) \geq UPC_{r,j} \; \forall\, j \in DELIVER, \quad r \in R_j, \quad i \in From_{r,j} - BASES : i \neq j, \quad t \in T_j, \quad s \in S_r \tag{29}$$

Equation (30) ensures the capacity limit is not violated at all times. Eq. (31) defines the initial payload weight at the beginning of the planning.

$$UPC_{r,j} \leq CAP_r \; \forall\, r \in RESOURCES, \quad j \in (To_r \cup InitPos_r) - BASES \tag{30}$$

$$UPC_{r,j} = UPC0_r \; \forall\, r \in RESOURCES, \quad j \in InitPos_r - BASES \tag{31}$$

4. Intersection

Novel constraints were developed to accommodate the *Intersection* tasks. Equation (32) sets up the value of $xor_{j,k}$ indicating if at least one of the tasks j with XOR relationship related to task k has been allocated. $k \in PotINTER$ and $j \in XOR_k$. Equation (33) ensures that for every task in $PotINTER$, no more than one related XOR task must be completed to avoid duplicate allocation.

$$\sum_{r \in R_j} \sum_{i \in From_{r,j}} \sum_{s \in S_r} \sum_{t \in T_j} x_{r,j,i}^{t,s} = xor_{k,j} \; \forall\, k \in PotINTER, j \in XOR_k \tag{32}$$

$$\sum_{k \in XOR_j} xor_{j,k} = 1 \; \forall\, j \in PotINTER \tag{33}$$

5. Preceded

Similar to the approach in [2], Eqs. (34), (35) and (36) are used to guarantee that *Preceded* tasks are allocated if and only if one of the preceding tasks is allocated.

$$\sum_{k \in PRECEDING_j} \sum_{r \in R_k} \sum_{i \in From_{r,k}} \sum_{s \in S_r} \sum_{t \in T_k} x_{r,k,i}^{t,s} \geq px_j \quad \forall j \in PRECEDED \tag{34}$$

$$\sum_{k \in PRECEDING_j} \sum_{r \in R_k} \sum_{i \in From_{r,k}} \sum_{s \in S_r} \sum_{t \in T_k} x_{r,k,i}^{t,s} \leq \sum_{k \in PRECEDING_j} l'_k \times px_j \quad \forall j \in PRECEDED \tag{35}$$

$$\sum_{r \in R_j} \sum_{i \in From_{r,j}} \sum_{s \in S_r} \sum_{t \in T_j} x_{r,j,i}^{t,s} = px_j \quad \forall j \in PRECEDED \tag{36}$$

Equations (37) and (38) ensures task $j \in PRECEDED$ is completed lag_j time slices after the preceding tasks

$$\sum_{r\in R_j}\sum_{i\in From_{r,j}}\sum_{s\in S_r} x_{r,j,i}^{t,s} = pxt_{j,t} \quad \forall j \in PRECEDED, t \in T_j \tag{37}$$

$$\sum_{r\in R_k}\sum_{i\in From_{r,k}}\sum_{s\in S_r}\sum_{t\in T_k:tk>t-lag_j-1} x_{r,j,i}^{t,s} \leq \left(1 - pxt_{j,t}\right) \ \forall j \in PRECEDED, k \in PRECEDING_j, t \in T_j \tag{38}$$

G. Base Constraints

Due to the required versatility of Bases nodes in the current formulation, novel constraints had to be developed to incorporate the specific role played by the base in any mission. The constraints required significant testing and modification before satisfactory operation was achieved. Bases are nodes, which must be visited by all resources at least once. Within a given plan, the planner needs to ensure all available resources have safely landed or were kept grounded at appropriate bases. Eq. (39) and (40) enforce resources landing during T_j time slices in which base j is available to accept vehicles.

$$\sum_{r\in R_j}\sum_{i\in From_{r,j}}\sum_{s\in S_r} x_{r,j,i}^{t,s} \leq card(RESOURCES) \quad \forall\, j \in BASES\,, \qquad t \in T_j \tag{39}$$

$$\sum_{j\in BASES}\sum_{t\in T_j}\sum_{i\in From_{r,j}}\sum_{s\in S_r} x_{r,j,i}^{t,s} \geq 1 \quad \forall\, r \in RESOURCES\colon InitPos_r \nsubseteq BASES \tag{40}$$

Resources must land before their fuel has been depleted hence Eq. (41) states that if a resource r is allocated to go from node i to a $BASES$ node j carrying sensor option s arriving at time t , it must arrive before running out of fuel at time $TR_{r,j}$.

$$t\, x_{r,j,i}^{t,s} \leq\ TR_{r,i} \ \forall\, r \in RESOURCES, j \in To_r, i \in From_{r,j} - BASES, t \in T_j, s \in S_r \tag{41}$$

In each planning instance the resource must terminate in a safe landing at an acceptable landing site. If resource r is allocated to any task (not a base) at time tj then it must be allocated to a base later on at time $tb \geq tj$. The use of $\geq$ in the Eq. (42) is to allow for vehicles to land as many times as they need during a certain mission to refuel or change sensor options.

$$\sum_{b\in B_r}\sum_{ib\,\in\, From_{r,b}}\sum_{tb\in T_b:tb\geq tj} x_{r,b,ib}^{tb,s} \geq x_{r,j,i}^{tj,s} \ \forall\, r \in RESOURCES, \qquad j \in To_r - BASES, \qquad i \in From_{r,j}, \qquad s \in S_r, \qquad tj \in T_j\colon i \neq j \tag{42}$$

H. Network Constraints

The number of time slices required to complete a certain task (l_j) are added to the travelling time in order to simplify the allocation. Equation (43) guarantees that at each time slice a resource can be allocated to at most one task.

$$\sum_{j\in To_r:t\in T_j}\sum_{i\in From_{r,j}}\sum_{s\in S_r} x_{r,j,i}^{t,s} \leq 1 \quad \forall\, r \in RESOURCES, \qquad t \in TIME \tag{43}$$

Equation (44) ensures that each resource departs from its initial position (assuming it is not a base) carrying the initial sensor combination defined by the operator.

$$\sum_{j \in To_r} \sum_{t \in T_j} x_{r,j,i}^{t,s} = 1 \quad \forall\, r \in RESOURCES, \qquad i \in InitPos_r - BASES, \qquad s \in InitSen_r \tag{44}$$

The first allocation must be physically feasible hence it cannot be before the time required for traveling from the initial position to the first task. Equation (45) prevents unfeasible first allocations.

$$\sum_{t \in T_j : t < TT_{r,j,i}} \sum_{s \in S_r} x_{r,j,i}^{t,s} = 0 \quad \forall\, r \in RESOURCES, \qquad i \in InitPos_r, \qquad j \in To_r : i \neq j \tag{45}$$

If a resource at k came from node j then it must have come from a certain node i to node j beforehand. Therefore, unless it is coming from an initial starting position, it must have a predecessor. If the intermediate node is not a base Eq. (46) ensures the sensor on-board the resource is retained. If the intermediate node is a base, then Eq. (47) allows the sensor on-board the resource to be changed.

$$\sum_{i \in From_{r,j} :\, if\ k \notin From_{r,k} \cup B_r\ then\ i \neq k} \ \sum_{tj \in T_j :\, tj + TT_{r,k,j} \leq tk} x_{r,j,i}^{tj,s} \geq x_{r,k,j}^{tk,s} \quad \forall\, r \in RESOURCES, \qquad k \in To_r,$$
$$j \in From_{r,k} - (InitPos_r \cup BASES), \qquad tk \in T_k, \qquad s \in S_r \tag{46}$$

$$\sum_{i \in From_{r,j} :\, if\ k \notin From_{r,k} \cup B_r\ then\ i \neq k} \ \sum_{tj \in T_j :\, tj + TT_{r,k,j} \leq tk} \ \sum_{sa \in S_r} x_{r,j,i}^{tj,sa} \geq x_{r,k,j}^{tk,s} \quad \forall\, r \in RESOURCES, \qquad k \in To_r,$$
$$j \in \left(From_{r,k} \cap B_r\right) - InitPos_r, \qquad tk \in T_k, \qquad s \in S_r \tag{47}$$

For a non-preemptive task, if a resource is at i, it can only go to one successor task j as assured by Eq. (48)

$$\sum_{j \in To_r :\, i \in From_{r,j}} \ \sum_{t \in T_j} \sum_{s \in S_r} x_{r,j,i}^{t,s} \leq 1 \quad \forall\, r \in RESOURCES,$$
$$i \in \left((To_r \cap NPre\,) \cup \ InitPos_r\right) - BASES \tag{48}$$

However, if the task is preemptive, the resource must not split between two tasks and hence the number of resources entering node j must be greater than or equal to the number of resources leaving as shown in Eq. (49).

$$\sum_{i \in From_{r,j}} \ \sum_{tj \in T_j :\, tj \leq tmax} x_{r,j,i}^{tj,s} \geq \sum_{k \in To_r} \ \sum_{tk \in T_k :\, tk \leq tmax + TT_{r,k,j}} x_{r,k,j}^{tk,s} \quad \forall\, r \in RESOURCES,$$
$$j \in (To_r \cap Pre), \qquad tmax \ \in TIME, \qquad s \in S_r \tag{49}$$

Similarly, at a base node j, if the resource is not starting from a base, we need to avoid the resource splitting at the base using Eq. (50). Nonetheless, if the resource is starting initially from the same base j, then Eq. (51) is used to avoid resource splitting at that base. The +1 in the left hand side of the inequality accounts for the initial position at that base. Finally, if the resource is initially starting from a base that is different from j, Eq. (52) is used to avoid resource splitting. Typical network constraints found in [48], [49],[14],[39] and [47] had to be modified to reflect the option of starting from different initial positions and accommodate the relationship with the Bases nodes.

$$\sum_{i \in From_{r,j}} \ \sum_{tj \in T_j : tj \leq tmax} \ \sum_{s \in S_r} x_{r,j,i}^{tj,s} \geq \sum_{k \in To_r : k \neq j} \ \sum_{tk \in T_k : tk \leq tmax + TT_{r,k,j}} \ \sum_{s \in S_r} x_{r,k,j}^{tk,s} \quad \forall\, r \in RESOURCES,$$
$$j \in B_r, \qquad tmax \ \in TIME \,:\, card(InitPos_r \ \cap BASES) = 0 \tag{50}$$

$$\sum_{i \in From_{r,j}} \ \sum_{tj \in T_j :\, tj \leq tmax} \ \sum_{s \in S_r} x_{r,j,i}^{tj,s} + 1 \geq \sum_{k \in To_r :\, k \neq j} \ \sum_{tk \in T_k :\, tk \leq tmax + TT_{r,k,j}} \ \sum_{s \in S_r} x_{r,k,j}^{tk,s} \quad \forall\, r \in RESOURCES,$$
$$j \in B_r \cap \ InitPos_r, \qquad tmax \ \in TIME \tag{51}$$

$$\sum_{i\in From_{r,j}} \sum_{tj\in T_j:\, tj\le tmax} \sum_{s\in S_r} x_{r,j,i}^{tj,s} \ge \sum_{k\in To_r:\, k\neq j} \sum_{tk\in T_k:\, tk\le tmax+TT_{r,k,j}} \sum_{s\in S_r} x_{r,k,j}^{tk,s} \quad \forall\, r \in RESOURCES,$$
$$j \in B_r - InitPos_r: card(InitPos_r \cap BASES) \neq 0, \qquad tmax \in TIME \tag{52}$$

I. Fuel Constraints

Inspired by the work in [47], novel fuel constraints were developed. $TR_{r,j}$ is the time at which fuel runs out for resource r when measured at task j . This value is carried forward from one non-base node to the other. When a resource visits a base the value gets updated based on current time, fuel time, take off time and any reduction in range due to payload options. Equation (53) ensures the value of the last allocable time slice $TR_{r,j}$ is carried correctly from one non base node to the other while Eq. (54) calculates a new value for the latest allocable time slice $TR_{r,j}$ when r is at a base. To incorporate resources that are already airborne, the initial remaining fuel capacity is taken into account using Eq. (55) where the remaining fuel at the initial position is set to $TR0_r$ provided that r is not starting from a base.

$$TR_{r,j} \le TR_{r,i} + \left(\left(1 - x_{r,j,i}^{t,s}\right)\times(Q + EndTime)\right) \ \forall\, r \in RESOURCES,$$
$$j \in To_r - BASES, i \in From_{r,j} - BASES : i \neq j, t \in T_j, s \in S_r \tag{53}$$

$$TR_{r,j} \le x_{r,j,i}^{t,s}\times\left(t + Q_r - TT_{r,j,i} - SR_{r,s}\right) + \left(\left(1 - x_{r,j,i}^{t,s}\right)\times(Q + EndTime)\right)$$
$$\forall\, r \in RESOURCES, j \in To_r - BASES, i \in From_{r,j} \cap BASES : i \neq j, t \in T_j, s \in S_r \tag{54}$$

$$TR_{r,j} = TR0_r \ \ \forall\, r \in RESOURCES, \qquad j \in InitPos_r - BASES \tag{55}$$

J. Sensor Selection

Mufalli et al. [3] use an optimization system that decides on the best combination of sensors to have on-board before starting any mission. In the present formulation, we extend that concept to allow the planner to plan a sensor selection schedule therefore allowing the resources to modify the installed sensor suite after the execution starts. The planner will choose the best sensor combination on each vehicle that fits individual phases of the generated plan. Eq. (56) ensures only one sensor combination is allowed to be carried by a given resource. A sensor combination can consist of many sensors as long as they do not exceed the resource sensor payload capacity which is determined by the pre-planner.

$$\sum_{s\in S_r} x_{r,j,i}^{t,s} \le 1 \ \ \forall\, r \in RESOURCES, \qquad j \in To_r, \qquad i \in From_{r,j}, \qquad t \in T_j \tag{56}$$

K. Finishing time

The finishing time of each resource r is calculated in Eq. (57) and the completion time of the entire mission plan calculated in Eq. (58). These times are multiplied by different weightings in the cost function to adjust the importance of individual finishing time in contrast to the importance of the overall mission completion time. Favoring individual finishing time will tend to reduce the number of resources used at the expense of overall finishing time. On the other hand, favoring completion time leads to shorter mission at the expense of greater use of a greater number of resources, which may have cost implications. Therefore the operator has the ability to change these settings according to his strategic goals.

$$Fintime_r \ \ge \sum_{t\in T_j} \sum_{s\in S_r} t\times x_{r,j,i}^{t,s} \quad \forall\, r \in RESOURCES, j \in To_r, i \in From_{r,j}: i \neq j \tag{57}$$

$$Comptime \ge Fintime_r \ \ \forall\, r \in RESOURCES \tag{58}$$

L. Cooperation

To ensure cooperation between different task types requiring different resource capabilities, Eq. (59) is used to enforce the allocation of cooperative tasks to be done at the same time slice.

$$\sum_{r\in R_{ja}} \sum_{i\in From_{r,ja}} \sum_{s\in S_r} x_{r,ja,i}^{t,s} = \sum_{r\in R_{jb}} \sum_{i\in From_{r,jb}} \sum_{s\in S_r} x_{r,jb,i}^{t,s} \ \forall\, k \in [1..Ntog], ja \in Together_k, jb \in Together_k : ja \neq jb, t \in T_{ja} \cap T_{jb} \tag{59}$$

M. Cost Function

The main objective is to identify a mission plan that meets all requirements at minimum overall cost. Since each platform and sensor combination offer different effectiveness in performing a certain task, then the importance of the effectiveness is dependent on the value (gain) of a certain task. To achieve this, each sub task must have a weighting assigned to it that relates the completion of the task (or part thereof) to the value accrued.

$$\alpha = \sum_{j\in TASKS} \sum_{r\in R_j} \sum_{i\in From_{r,j}} \sum_{t\in T_j} \sum_{s\in S_r} x_{r,j,i}^{t,s} \times (c_1 \times (PE_{r,j} \times SE_{s,j}) G_j - (c_2 \times (SUC_s + MC_r))$$

The weighting c_1 and c_2 are used to vary the importance of effectiveness versus cost. Increasing c_1 would place more emphasis on selecting the most effective resource for each task. Increasing c_2 would increase the effect of the cost in the objective function.

Secondary objectives include reducing take-off, landing and additional operational cost. This allows the resource allocator to, whenever possible, chose a base with low fees for the vehicle. The weighting c_3 is used to amplify the Take-Off and Landing cost in order to reduce unnecessary landings during mission execution. β represents the weighted sum of the landing costs and γ represents the weighted sum of the take-off costs and the additional operational costs. β and γ should be minimized.

$$\beta = \sum_{r\in RESOURCES} \sum_{j\in B_r} \sum_{i\in From_{r,j}} \sum_{t\in T_j} \sum_{s\in S_r} x_{r,j,i}^{t,s} \times c_3 \times LC_{r,j}$$

$$\gamma = \sum_{r\in RESOURCES} \sum_{i\in B_r} \sum_{j\in To_r:\, i\neq j} \sum_{t\in T_j} \sum_{s\in S_r} x_{r,j,i}^{t,s} \times c_3 \times (TOC_{r,j} + AOC_r)$$

The variable δ represents the weighted traveling time between nodes that should be minimized.

$$\delta = \sum_{r\in RESOURCES} \sum_{j\in To_r} \sum_{i\in From_{r,j}} \sum_{t\in T_j} \sum_{s\in S_r} x_{r,j,i}^{t,s} \times c_2 \times TT_{r,j,i} \times MC_r$$

The variable ε represents the weighted sum of the finishing time of each resource. ε should be minimized. This will also reduce the waiting time before each task. c_2 is included in ε in order to reduce the cost incurred due to waiting before tasks.

$$\varepsilon = \sum_{r\in RESOURCES} c_2 \times c_4 \times Fintime_r$$

The variable θ represents the weighted value of the total mission time that should also be minimized.

$$\theta = c_5 \times Comptime$$

The weightings c_4 and c_5 are used to vary the relative importance of reducing individual completion time versus overall completion time.

The weighted sum of the remaining fuel represented by ι should be maximized.

$$\iota = \sum_{r \in RESOURCES} \sum_{j \in (To_r \cup InitPos_r) - BASES} c \times TR_{r,j}$$

The weight c is a very small number(<0.01) that reduces the contribution of the fuel value into the cost function while ensuring the remaining fuel is maximized.

The weighted sum of the payloads carried represented by κ should be minimized.

$$\kappa = \sum_{r \in RESOURCES} \sum_{j \in (To_r \cup InitPos_r) - BASES} c \times UPC_{r,j}$$

The variable λ is used to minimize the number of $w_{j,t}$. Minimizing λ ensures the correct number of time slices is allocated for *Loiter* tasks.

$$\lambda = \sum_{j \in LOITER} \sum_{t \in T_j} c \times w_{j,t}$$

The final cost function is:

$$Max: \ \alpha - \beta - \gamma - \delta - \varepsilon - \theta + \iota - \kappa - \lambda \tag{60}$$

The operator can rank preference of utility by giving the coefficients c_1, c_2, c_3, c_4, c_5 a value between 1 and 100. The ranking and the differences between the values of the coefficient describes the relative importance of time, cost and effectiveness to the operators.

The ILP formulation of the resource allocation model, was solved using the commercially available solver IBM ILOG CPLEX 12.6.1[50]. All the simulations were carried out on a desktop computer running Windows7 with a 3.20GHz Intel Core i7 3930K processor and 32 GB of RAM. The formulation can be solved using alternative solution methods, e.g. Genetic Algorithms or used as a basis for the development of near optimal heuristics algorithms that may reach solutions more quickly than CPLEX at the expense of solution optimality.

N. Main Features

The proposed ILP model offers a number of standard features such as planning with time windows, planning for heterogeneous tasks and heterogeneous resources, accommodation of inter task dependencies and restriction of fuel capacities. The contribution of the model resides in the holistic approach to the resource allocation problem, allowing the model to be used in diverse civilian applications. *Transportation* and *Intersection* tasks are two new capabilities not considered in [2] [27]. Transportation is very common and crucial for a civilian UAV sharing system and constitutes one of the main services that needs to be present. The implementation of transportation services is also novel in that it allows the UAVs to exchange cargo at bases. Cargo transshipment allows heterogeneous resources to collaborate on transportation requests. *Intersection* tasks offer the ability to combine multiple requests coming from unrelated users into one request therefore reducing the cost and time of servicing those requests. Moreover, the sensor selection feature is a major contribution allowing the model to also select appropriate sensors for each resource similar to [3]. The ILP model also offers an additional feature that allows it to plan sensor changes during execution by landing at a base and taking off afterwards to continue the mission.

The base in this model is not just a location node where resource can travel to; it offers logistics services to those resources such as transshipment, sensor change and refueling together with the typical landing and take-off service. The base also acts as a waiting node where the resources do not incur any cost over time.

In order to accommodate all the new features with all the standard features in one system the formulation used a different approach to represent remaining fuel and refueling constraints. It also had to use an approach to payload capacity similar to the one used in delivery truck problems in order to allow the capacity to increase and decrease according to the transported cargo.

Our main application space is a multi-UAV services sharing system, therefore the design of the objective function was very important to capture the possible operator needs and preferences. Combining objectives from different categories such as time, cost and reward into a meaningful value is achieved through converting all the objectives into a unified currency before being used in the optimization. It includes cost of sensor change, take-off, landing, travel time and a gain of each task weighted by a resource effectiveness factor for performing the task. In addition, the operator can use the coefficients c_1, c_2, c_3, c_4, c_5 to define a preference for the optimization goals between mission cost, mission completion time and mission effectiveness.

V. Results

A. Overview

The results section is divided into two main sections. Subsection B presents the evaluation of results from a set of detailed numerical experiments on a single scenario that explores the effect of optimization choices on mission outcomes for this single scenario. Subsection C presents results from experiments on multiple different scenarios that explores the performance of the approach in different scenarios of increasing complexity.

B. Baseline Scenario

1. Scenario description

To illustrate the resource allocation system behavior in a real context, we first take a single scenario with 4 resources and 7 requests shown in Fig. 4: A vehicle service provider is offering users the use of its fleet of autonomous vehicles. Users send their requests to the centralized vehicle management center where the allocation process takes place. Once a plan is generated, resources are autonomously allocated to the appropriate user and the service begins. Upon receipt of any new requests, the resource allocation model is solved using the current states of resources in order to generate an up to date plan. This scenario incorporates a sample of each type of task requests and uses a combination of heterogeneous vehicles to service the requests in an efficient way. We will refer to this scenario as the baseline scenario.

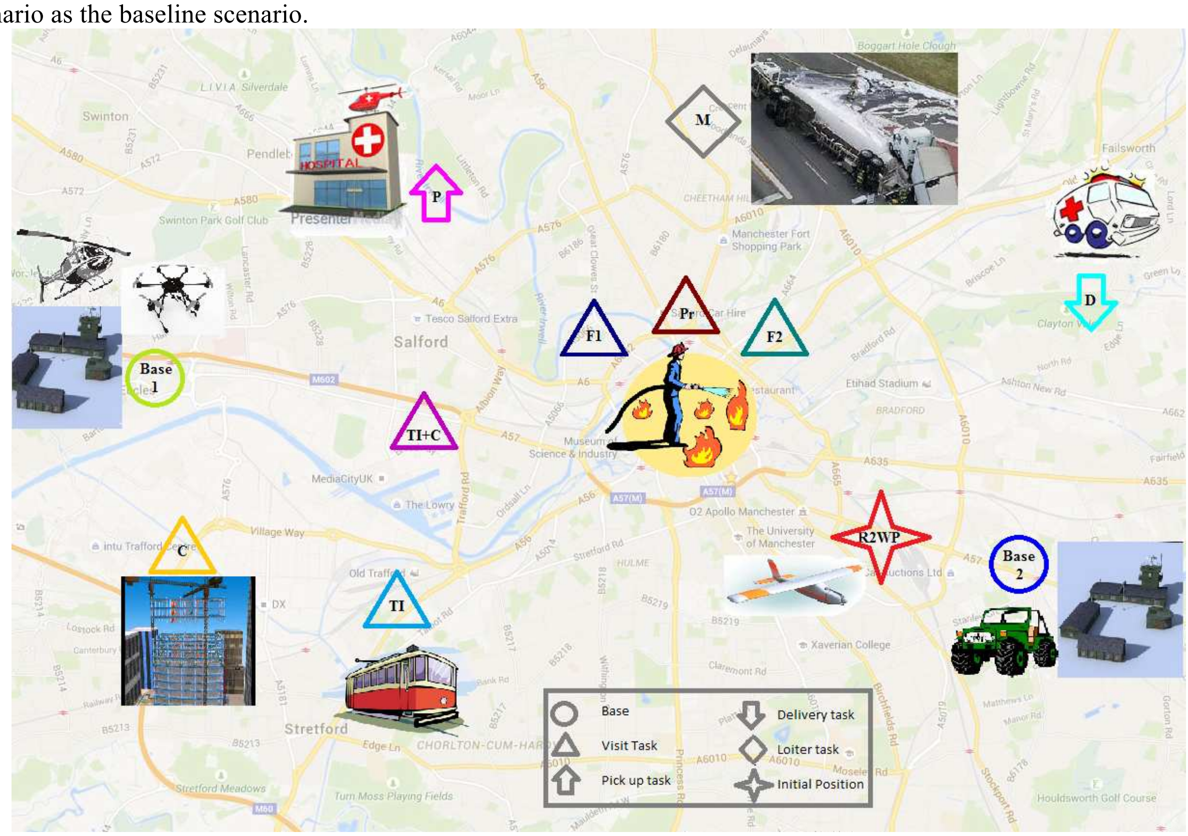

Fig. 4: Illustration of the baseline scenario with relative location between tasks, bases and resources.

An emergency medical team requires a specialist medical kit. They contact the nearest hospital who send a request through the local vehicle services center for a point to point pick-up and delivery service. Due to an earlier truck accident, the medical team on the ground is only accessible by air hence the final delivery must be done by a UAV. The accident management team dealing with the truck accident has already sent a request for an aerial assessment of the area. On the other side of the town, the local tram operator is performing maintenance on a line and is requesting an aerial inspection of the tracks while a nearby construction manager is asking for his daily site inspection to monitor the progress of his work.

Finally, a fire is developing in a warehouse downtown and the fire department requires simultaneous Visual and IR scans to make sure the fire is being appropriately contained. The fire department is also asking for a second IR scan 5 minutes after the first one to monitor the progress of the fire.

Table 1 shows the description of the resources available with their current status. We notice that all resources are grounded at different bases except the fixed wing aircraft which was already airborne and is currently at R2WP. Table 1 shows the sensor options available and the requests sent by various users in the local area. The time windows are intentionally chosen to be wide in order to give the resource allocation some freedom in moving tasks around. For narrow time windows, the solution space is very constrained and there is little benefit in optimizing the mission. We thus choose a planning horizon of 25 minutes based on the latest possible servicing time requested. The planning horizon is divided into 25 time slices hence each time slice in the plan is equivalent to 1 minute in real life. Note that the planning time resolution can be adjusted depending on the density of tasks in the planning horizon and the available time for planning. High time resolution increases the size of the ILP model and hence increases solution time.

2. Results with default allocation parameters

The schedule presented in Fig. 5 shows the changes in states of each resource with the passage of time. The first graph (Fig. 5a) shows the schedule of each vehicle. The x axis represents time slice number, while each block represents the state which the vehicle is in for that period of time. The light grey blocks represent the time spent traveling between nodes, including the time taken to take off and refuel if the vehicle is leaving a Base node and the time it takes to land if the vehicle is arriving at a Base node. Empty spaces between blocks in the schedule represent free allocable time. The model coding is such that if the vehicle was at a base before being allocated free time, it remains there until it is required to move, otherwise if it was airborne it loiters while waiting to continue its mission. The colors denote different tasks and relate the illustration to the schedule. The second graph (Fig. 5b) shows the sensor combinations used to perform each task. The exchange of sensor options at bases can be correlated with the schedule in Fig. 5b.

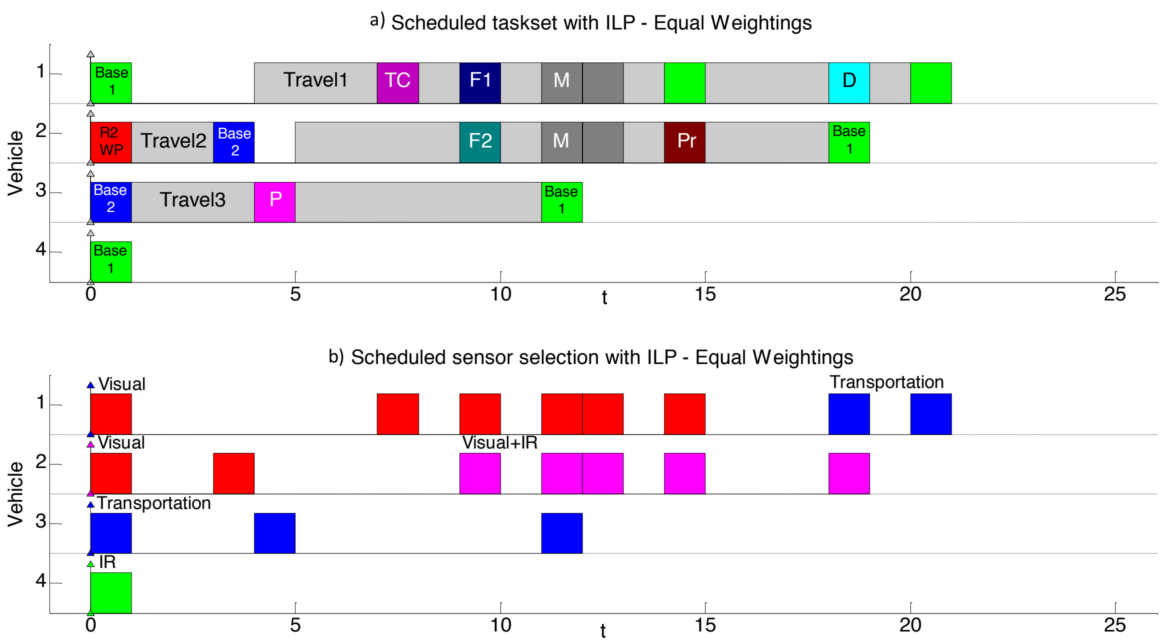

Fig. 5: Sample schedule results for the baseline scenario.

Table 1: Resources, Sensor Combinations and Tasks used in the baseline scenario.

Sensor Combinations

Sen Combo #	Type	Weight	Usage cost (£/Time Slice)
1	Visual Sensor (Vis)	1	2
2	Infrared Sensor (IR)	3	5
3	Transportation Bay (Tran)	0	0
4	Visual Sensor + Infrared Sensor	4	7

Resources

Res #	Type	Speed (m/s)	Sensors Compatible	Sensor onboard	Sensor Capacity (no off)	Payload Capacity (Kg)	Maintenance Cost(£/Time Slice)	Initial Position	Remaining Flight Time
1	MultiRotor	15	Vis,IR,Tran	Vis	1	5	3	Base 1	20
2	FixedWing	10	Vis, IR	Vis	2	10	5	R2WP[a]	8
3	Rover	5	Vis, Tran	Tran	1	15	2	Base 2	20
4	Helicopter	15	IR,Tran	IR	1	5	10	Base 1	20

Task details

Description	Symbol	Type	Sensor	Platform	Time Window	Execution Time	Gain £
Pick up emergency equipment weighing 1Kg from the hospital	P	Visit	Tran	UGV	[0-6]	1	20
Deliver emergency equipment to first responders	D	Deliver	Tran	VTOL	[3-20]	1	20
Provide aerial view of tanker spill accident to crew on the ground	M	Loiter	Vis	Airborne	[10-15]	2	100
Aerial inspection of construction site	C	PotInter	Vis	Airborne	[2-18]	1	40
Aerial inspection of tramline	TI	PotInter	Vis	Airborne	[5-20]	1	100
Intersection of TI and C	TC	Inter	Vis	Airborne	[5-18]	1	140
Visual Fire Inspection	F1	Visit	Vis	Airborne	[1-8]	1	100
IR Fire	F2	Visit	IR	Airborne[b]	[1-8]	1	100
Fire progress inspection	Pr	Preceded	IR	Airborne	[2-14]	1	100

[a] RxWP is used to indicate the airborne initial waypoint position of resource x.
[b] Platform Effectiveness in performing TaskF2 is: Vehicle1 (0.1) , Vehicle2 (0.1), Vehicle3 (-1) and Vehicle4 (1). Platform Effectiveness for all other tasks is (1) for capable platforms and (-1) for incapable platforms.

In the presented scenario, the resource allocator uses equal weights between mission cost, mission completion time and mission effectiveness. The coefficients in the cost function are $c_1 = c_2 = c_3 = c_4 = c_5 = 1$. The resource allocator decided to use three resources only, since there is a cost for utilizing more resources than needed. Vehicle1 is tasked to perform TaskTC at t=7, instead of performing TaskTI and TaskC individually. In this case, it is cheaper to perform the *Intersection* task by saving the cost of traveling between the individual tasks while retaining the same service quality. Once TaskTC is completed, Vehicle1 flies to TaskF1 at t=9 in order to collaborate with Vehicle2 to perform both TaskF1 and TaskF2 simultaneously, each with a different sensor. This collaboration is then followed by cooperation between Vehicle1 and Vehicle2 to performing TaskM from t=11 till t=13. Having two resources serving TaskM provides an added value to the quality of the service which, in this scenario, is providing information for accident and spillage assessment. Vehicle1 lands at Base1 at t=14 in order to switch its payload, pick-up the emergency medical kit that was brought to Base1 by Vehicle3 and deliver it to TaskD at t=18. Finally, Vehicle1 lands at Base1 marking the end of its schedule. The time it takes to take off and service Vehicle1 is accounted for in the traveling time after each base.

The resource allocator found it efficient to land Vehicle2 at Base2 on t=3 in order to change its sensor, refuel and then take off at t=5. Note that the optimal identified solution uses Vehicle2 to carry both Visual and IR sensors, leading to a higher operations cost, in order to serve TaskF2, TaskM and TaskPr at t=9, t=11 and t=14 respectively.

Since the pick-up task from the hospital (TaskP) can only be completed by a ground vehicle, it is automatically allocated to the only available option Vehicle3. Vehicle3 leaves Base2, where it was located, to pick up the emergency kit and deliver it to Base1 where it will later be taken by Vehicle1 to the medical team on the ground. The model found it more efficient (less costly) to utilize Vehicle1 for the delivery task TaskD than to use Vehicle4. This is in spite of the need to perform the landing of Vehicle1 and exchange payloads before it can be used for the delivery.

The resulting solution utilized an average of 59% of the available resources. This percentage is calculated based on the ratio of flight time to the potential flight time of each resource. The potential flight time of each resource is given by $\left(\left(\frac{Maximum\ energy\ capacity_r}{Energy\ consumption\ per\ unit\ of\ time_r}\right) - Logistics\ time_r\right) \times Number\ of\ bases\ visits_r$ with $Logistics\ time_r = \sum_{All\ base\ visits} TOC_{r,b} + LC_{r,b}\ \forall\ r \in RESOURCES$. Approximately 30% of the flight time was used to execute tasks while the rest is distributed between traveling and waiting. Finally, the benefit to cost ratio is 2.34:1 which indicates a positive profit under the cost assumptions taken. The benefit to cost ratio of the mission is the total gain obtained by all resources divided by the total cost incurred by all resources. Mission benefit to cost ratio is given by $\frac{\sum_{\forall\ r \in RESOURCES} Total\ Gain_r}{\sum_{\forall\ r \in RESOURCES} Total\ Cost_r}$.

3. Results for the variation of allocation parameters to achieve different operational goals

Taking the same scenario, we change some of the main parameters in the allocation model to assess its ability to accommodate different operational goals. Figure 6 presents four solutions to the baseline scenario. Figure 6a is the same as the solution in Fig. 5 where the weightings between the three main operational goals are equal. This is referred to as "solution 1".

The solution in Fig. 6b, referred to as "solution2", considers maximizing effectiveness to be more important. We use the following coefficient set in the cost function: $c_1 = 100,\ c_2 = c_3 = c_4 = c_5 = 1$. The main difference from solution1 is the allocation of TaskF2 to Vehicle4, which is the most effective in performing that task. All other tasks are rescheduled accordingly. Hence, TaskF1 is performed before TaskTC in order to ensure collaboration between Vehicle1 and Vehicle4 during the same time slice on TaskF1 and TaskF2. The remainder of the plan is the same as solution1.

The solution in Fig. 6c, referred to as "solution3", considers minimizing operations cost to be more important. We use the following coefficient set in the cost function: $c_2 = 100,\ c_1 = c_3 = c_4 = c_5 = 1$. The resource allocator attempts to reduce the operations cost by utilizing the plan in solution1 and eliminating the cooperation on TaskM. It is cheaper for the allocation to use the less effective Vehicle2 in performing TaskF2. It is also cheaper not to dispatch two resources to attend one task.

The solution in Fig. 6d, referred to as "solution4", considers completion time to be more important. We use the following coefficient set in the cost function: $c_5 = 100,\ c_1 = c_2 = c_3 = c_4 = 1$. We can immediately identify one main difference which is the use of Vehicle4 in performing TaskD and the use of Base2 for transshipment instead of Base1. For Vehicle1, TaskTC was moved from the beginning of the plan allowing TaskF1 and TaskF2 to be completed at an earlier time. Vehicle4 collaborated with Vehicle1 on TaskF1 and TaskF2, so that Vehicle2 can subsequently attend TaskPr after the lag of 4 time slices between TaskF1 and TaskPr. TaskM was attended by Vehicle1 alone in order to allow for the reduction in the schedule timeline.

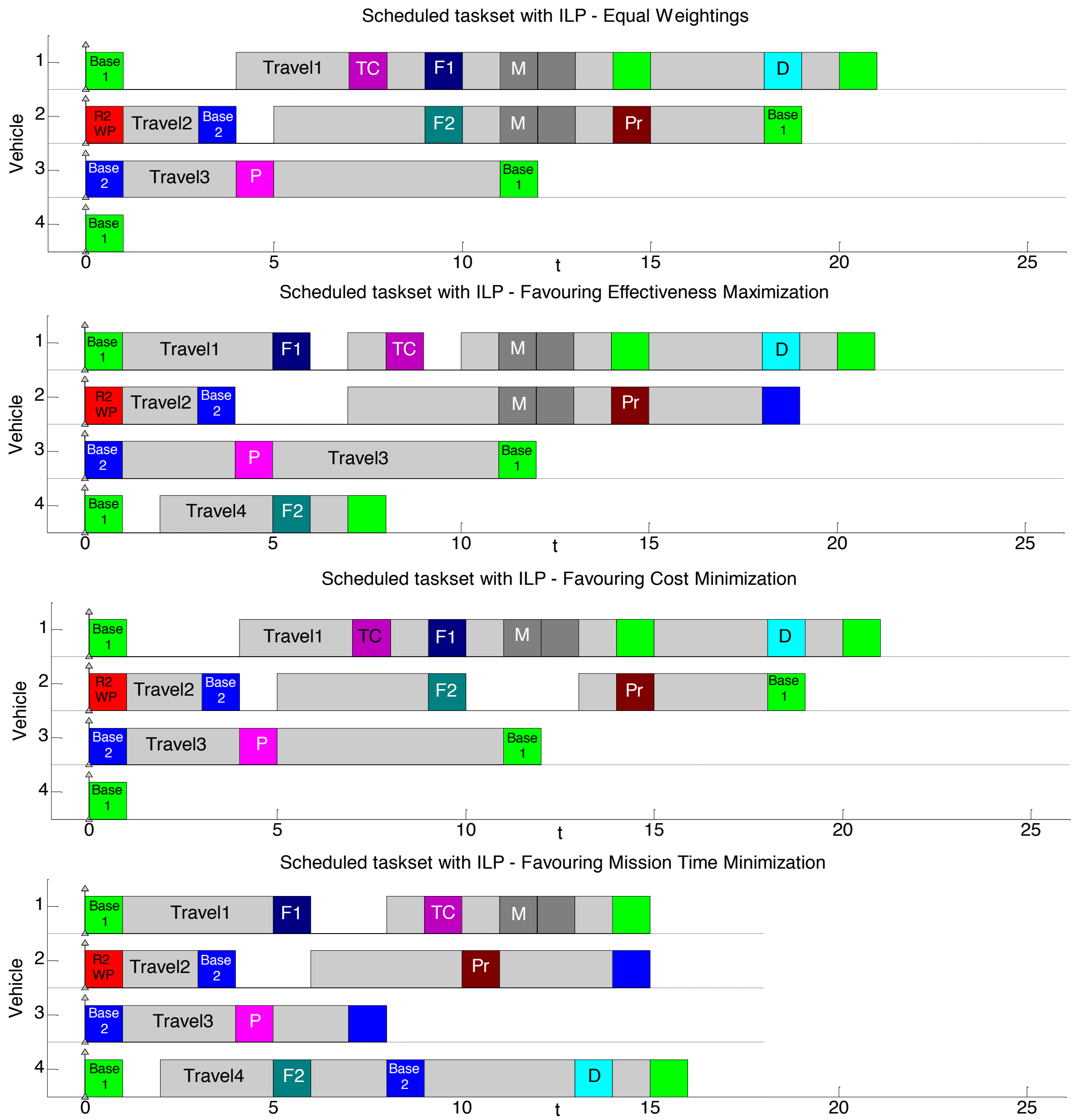

Fig. 6: Effect of changing optimization goals on mission solutions for the baseline scenario.

Figure 7, 8 shows a comparison between the four different solutions presented in Fig. 6. The StdDev is the standard deviation of the resource loading in each solution. A higher value for this metric, indicates a more unfair (unequal) resource loading.

In solution2, the increase in effectiveness was based on the mobilizing of an additional, more effective resource which leads to an increase in cost and a reduction in benefit to cost ratio in comparison with solution1. Solution 2 has the worst load distribution because Vehicle3 and Vehicle4 are lightly loaded compared to Vehicle1 and Vehicle2. This is also reflected in the low percentage of resource utilization (50%).

In solution3, the resource allocator reduced the operations cost by reducing a collaboration on a task that deprived the mission from extra benefit. The resulting benefit to cost ratio is low (1.82:1) compared to solution1 (2.34:1) and solution2 (1.97:1). The resource allocator ended up with a relatively fair load distribution amongst mobilized resources and has the second best percentage of resource utilization (57%). However, the mission plan also is disadvantaged by long waiting times indicated by the lowest percentage of flight time used for execution amongst all four solutions (24%).

In solution4, the allocation model utilized a greater number of different resources to reduce the mission completion time by 25%. This lead to the fairest distribution of load between all four solutions but at a lower benefit to cost ratio. The overall benefit (720) was less than solution1 (866) and solution3 (920) however the overall cost was also the highest amongst all four solutions (479). This is a clear trade-off between time and cost.

The benefit to cost ratio dropped in solution2 to 1.97:1 compared to solution1 (2.34:1) due to the extra cost associated with operating Vehicle4. The drop was not high since the gain obtained from the tasks partially compensates for the extra cost added by Vehicle4 operation. The ratio in solution3 dropped to 1.82:1 due to a larger drop in overall benefit versus overall cost in comparison to values of solution1. The benefit to cost ratio dropped significantly in solution4 to 1.5:1 indicating severe losses in terms of profit compared to the advantage of finishing 25% earlier than the other three solutions.

Finally, the time it took to calculate solution2 is 9 seconds, which is less than the 10 seconds taken by solution3, the 24 seconds taken by solution1, and much less than the 64 seconds taken by solution4. Figure 7 shows the comparison between computation times. This indicates that when more constraints are present the less feasible options the allocation system has to consider and the quicker it can produce an optimal solution.

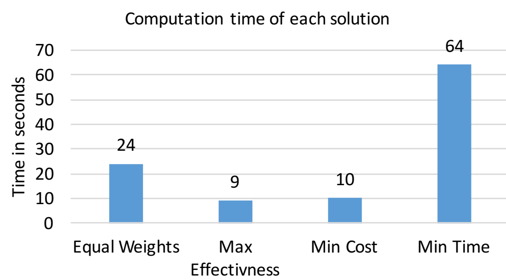

Fig. 7: Effect of varying mission goal on solution computation time, baseline scenario.

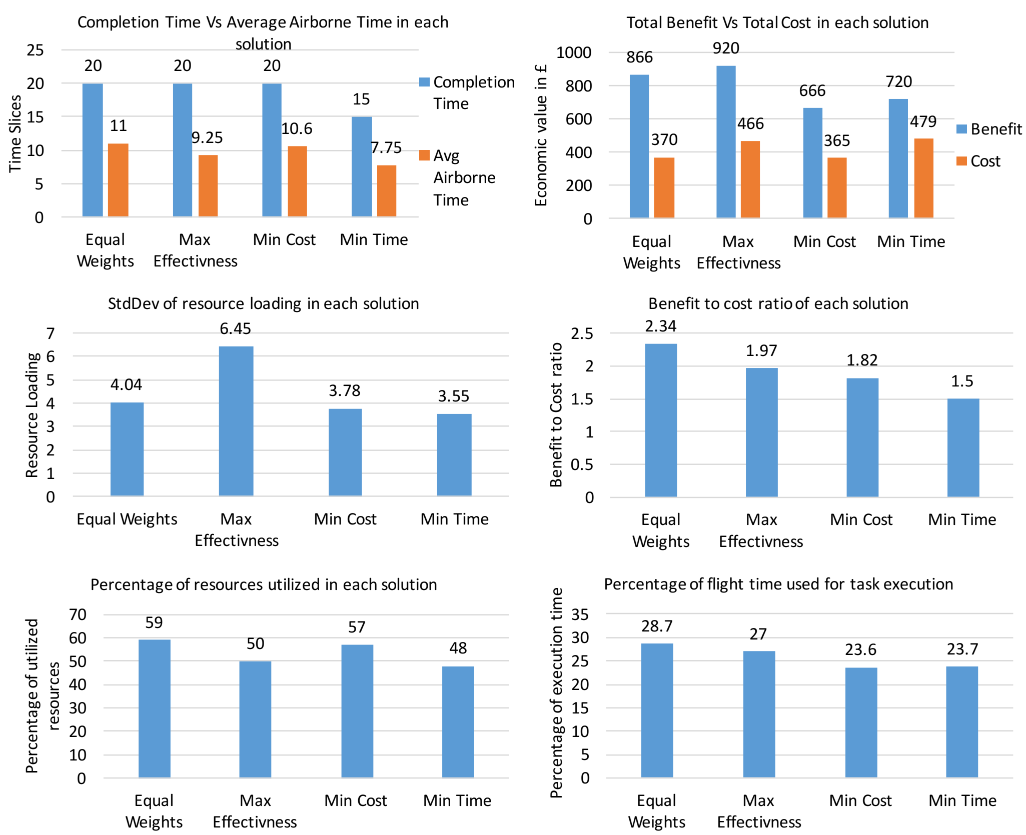

Fig. 8: Effect of mission objective on mission performance metrics, baseline scenario.

4. Demonstration of dynamism

The presented model has the capability to dynamically accommodate new requests during execution. Resources are able to start from their current states defined by their current locations, remaining fuel capacity and the currently installed sensor combination. This allows the model to be re-solved at any point during execution for the remaining tasks. The reallocation is triggered by specific events such as the addition, removal and modification of tasks, and the unexpected loss of resources. Limiting the events that can invoke a reallocation greatly reduces the possibility of churning cases. Churning is usually an issue in dynamic real time multi-agent systems where the ability to reallocate tasks can lead to repeated reassignment. However, this is not an issue in the present model due to the limited causes of reallocation and their low frequency of occurrence. On the other hand, plans may get modified after every allocation cycle, leading to a weak load distribution in certain very dynamic scenarios. However, the produced plan in each allocation cycle would still be the best outcome based on the situation at that moment in time.

To demonstrate dynamism, we use the baseline scenario presented in Fig. 4 and the mission parameters that resulted in the plan in Fig. 5, with the exception of the time at which the requests are received. The resource allocation system receives two sets of requests one at time t=0 and the other at time t=10 into the execution. We take the requests sent by the fire department to be the delayed requests hence the first set of requests include Tasks P, D, M, TI and C detailed in Table 1 while the second set arriving at t=10 includes Tasks F1, F2 and Pr.

Fig. 9 shows the plan for the first set of requests in the new scenario which looks very similar to the one in Fig. 5 but without the fire department requests. The time taken to generate this plan was 5 seconds. The plan utilizes 49% of available resources at the time, 26% of the flight time is being used to execute tasks and the benefit to cost ratio is 1.97:1.

The plan starts execution at t=0 and by t=10 TaskTC and TaskP have been completed, Vehicle1 and Vehicle2 are flying towards TaskM, the emergency cargo in Vehicle3 is on its way to be delivered to Base1 and Vehicle4 is still at Base1. At t=10 the second set of requests is obtained, and the resource allocation system loads all pending tasks together with the newly received requests into the allocation model. The current positions of Vehicle1, Vehicle2 and Vehicle3 are loaded into the allocation model as R1WP, R2WP and R3WP.

Fig. 10 shows the plan for the second set of requests combined with the pending tasks from previous set. It took 10 seconds to generate this plan. The initial time t=0 in this plan is equivalent to t=10 in the first plan. The plan utilizes 50% of available resources at the time, 26% of the flight time is being used to execute tasks and the benefit to cost ratio is 1.85:1. With the receipt of TaskF2 requiring a flying vehicle equipped with IR, the allocation system decided to mobilize Vehicle4 to attend TaskF2 and collaboration with Vehicle2 attending TaskF1. Vehicle1 continued its plan as scheduled. Vehicle4 has the IR sensor on-board and is tasked to attend TaskPr which due to the precedence constraint with lag time forced Vehicle4 to wait in the air for 3 time slices. Vehicle3 continued its plan, without any change, delivering the emergency medical kit to Base1 for where it will later be picked up by Vehicle1 for final delivery.

Comparing the result of the dynamic scenario presented in Fig. 9 and Fig. 10 to the optimal result that used prior knowledge of all tasks presented in Fig. 5, we can notice a reduction in the plan quality. When the tasks arrived in two sets, it took longer overall to execute them: 23 time slices in the dynamic scenario vs 20 time slices in the optimal case. More time was wasted in waiting and more cost incurred due to increase in resource utilization. The total mission benefit to cost ratio decreased to 2.08 in the dynamic scenario compared to 2.34 in the optimal case. This is an 11% reduction from optimal. While the same gain was achieved (866) in both cases, the cost in the dynamic scenario increased to (416) from (370). It is expected that a more efficient and more cost effective plan will be generated if all requests are known prior to the allocation. However, in reality and due to the nature of service this system is intended to provide, it is evident that requests are going to be submitted at unpredictable times, and therefore it is crucial that the allocation model is able to successfully cope with the dynamism imposed by user request.

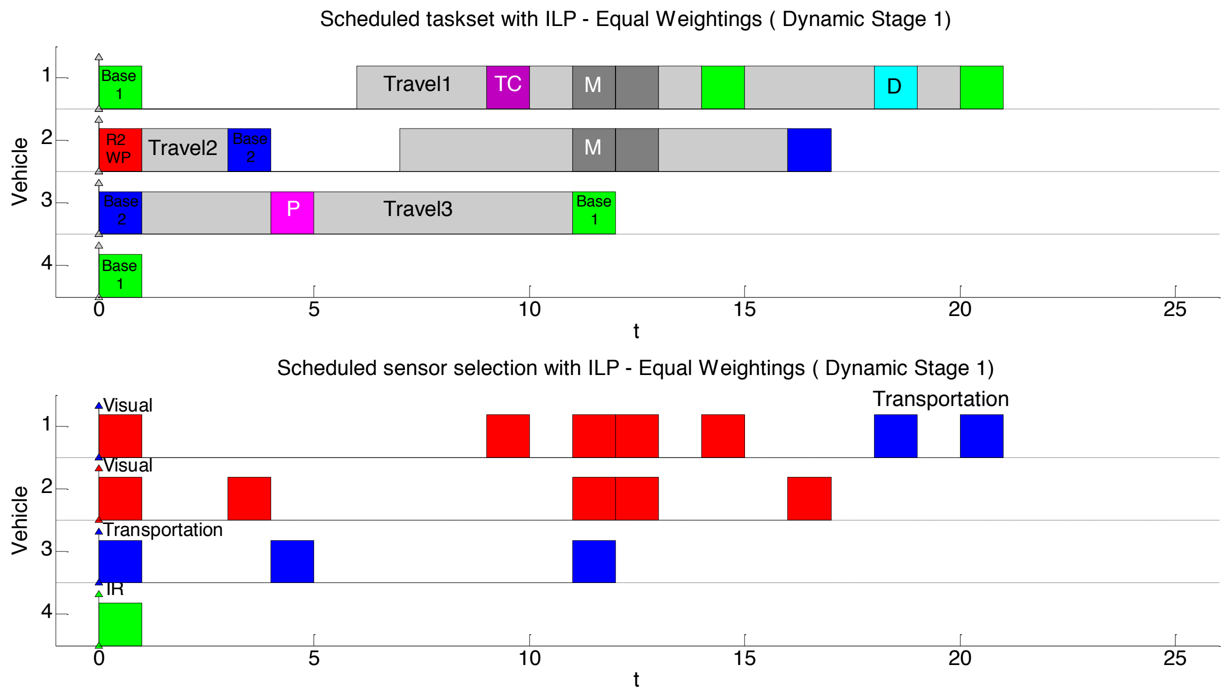

Fig. 9: Schedule of the requests received at t=0.

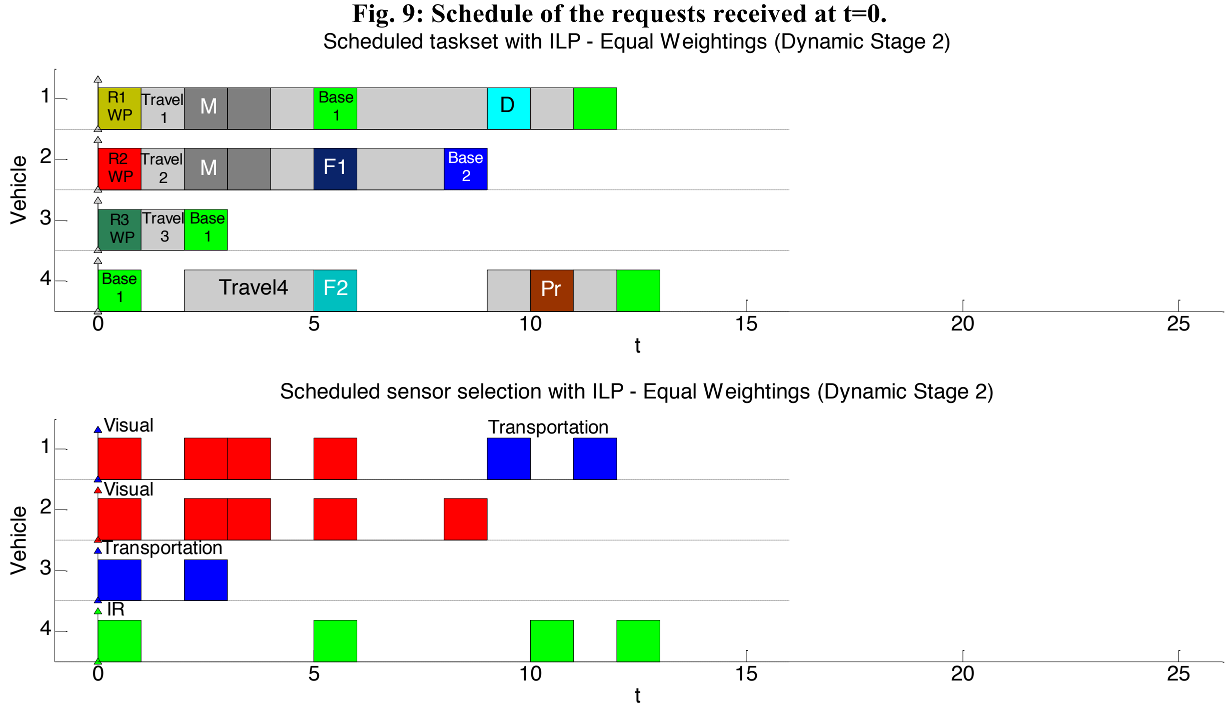

Fig. 10: Schedule of the additional requests received at t=10.

C. Performance Study

1. Simulation description

This section presents the evaluation of the performance of the resource allocation system as the scope and size of the mission scenario is varied. These sets of missions will be referred to as the performance study sets.–Three performance studies were carried out all sharing the same environment described in Table 2. The studies considered the effect of both the time window size and the number of time slices used in a simulation, the relative performance compared to other allocation methods and the relative performance at limited computation time. Random tasks were generated in random locations within a 6.3 Km^2 area. Five task parameters are randomly generated from the list of options shown in Table 2: location, type, starting time, required platform and required sensor.

Table 2: Simulation environment setup for the performance study sets.

Sensor Combinations

Sen Combo #	Type	Weight (Kg)	Usage cost (£/Time Slice)
1	Visual Sensor (Vis)	2	0.2
2	Infrared Sensor (IR)	2	0.2
3	Transportation Bay (Tran)	0	0

Resources

Res #	Type	Speed (m/s)	Sensors Compatible	Sensor on-board	Sensor Capacity	Payload Capacity (Kg)	Maintenance Cost (£/Time Slice)	Initial Position	Remain-ing Flight Time (min)
1	Multi-Rotor	10	Vis, IR, Tran	Vis	1	3	0.2	Base 1	24
2	Fixed Wing	15	Vis, IR, Tran	Vis	1	10	0.5	Base 1	24
3	Hybrid VTOL	10	Vis, IR, Tran	Vis	1	3	0.2	Base 1	24

Randomized task parameters

Parameter	Range
Location	Latitude ∈ [53.462226, 53.484978] Longitude ∈ [-2.257769, -2.219175] Altitude ∈ [10m,120m]
Starting time	[0, (Planning horizon – Time window size)]
Required platform	Uniformly randomly distributed amongst all available types shown in Table 3
Required sensor	Uniformly randomly distributed amongst all available types shown in Table 3
Task type	Randomly chosen with a predefined percentage of occurrence from [Visit, Loiter, Transport, Precedence, XOR]

A number of random tasks are grouped to form a mission. Each performance study contains a number of mission sets. Table 3 presents the simulation parameters and characteristics that are fixed amongst all sets and parameters that vary between those sets. Each set contains 14 randomly generated missions based on a combination of the variable parameters. Limits on parameter value ranges were set based on experience of what typically produced operationally meaningful results.

Table 3: Options for the parameters varied to generate simulation case studies for performance study sets.

Parameter	Effect of TW and NTS	Comparison of allocation methods	Effect of limiting computation time to 120 seconds
Simulation characteristics that vary between each performance study set			
Allocation Method	[ILP]	[ILP CBBA Greedy]	[ILP]
NT/NR	[3 5]	[4]	[3 5]
NR	[2 3]	[2 3 4]	[2 3]
Time window size in minutes (TW)	[4 6 8 10]	[5]	[4 6 8 10]
Number of time slices (NTS)	[20 40 60 80 100]	[50]	[20 40 60 80 100]

Simulation characteristics fixed over all performance study sets

Parameter	Effect of TW and NTS	Comparison of allocation methods	Effect of limiting computation time to 120 seconds
Percentage of task types in all mission sets	Visit: 30, Loiter: 30, Transport: 15, Precedence:15, XOR: 10	Visit: 50, Loiter: 50	Visit: 30, Loiter: 30, Transport: 15, Precedence:15, XOR: 10
Number of missions per set	14	10	14
Planning horizon in minutes	40	25	40
Optimization coefficients	Equal weights	Equal weights	Equal weights
Task reward	£100	£100	£100
Platform types	NR=2: Res1 x1, Res2 x1 NR=3: Res1 x1, Res2 x1, Res3 x1 NR=4: Res1 x2, Res2 x2	NR=2: Res1 x1, Res2 x1 NR=3: Res1 x1, Res2 x1, Res3 x1 NR=4: Res1 x2, Res2 x2	NR=2: Res1 x1, Res2 x1 NR=3: Res1 x1, Res2 x1, Res3 x1 NR=4: Res1 x2, Res2 x2
Sensor types	Vis, IR, Tran	Vis	Vis, IR, Tran
Computation time limit in sec	7200 (2hrs)	7200 (2hrs)	120 (2min)

2. Study of the effect of task time window size (TW) and number of time slices (NTS)

In order to study the effect of varying the time window size and the time resolution for different numbers of resources, we considered 80 different sets, each comprised of 14 random missions. A summary of the results is presented in Fig. 11 and Fig. 12. Note that each data point is obtained from taking an average from the 14 missions in each set for each parameter combination. The results are organized such that a given row of plots in a figure corresponds to a test case with a specific Number of Resources (NR) and task to resource ratio. The different lines in each plot are for different number of time slices (increasing number means increasing time resolution). The x axis in each plot is the time window for the tasks, where increasing time window corresponds to a less restrictive scheduling requirement.

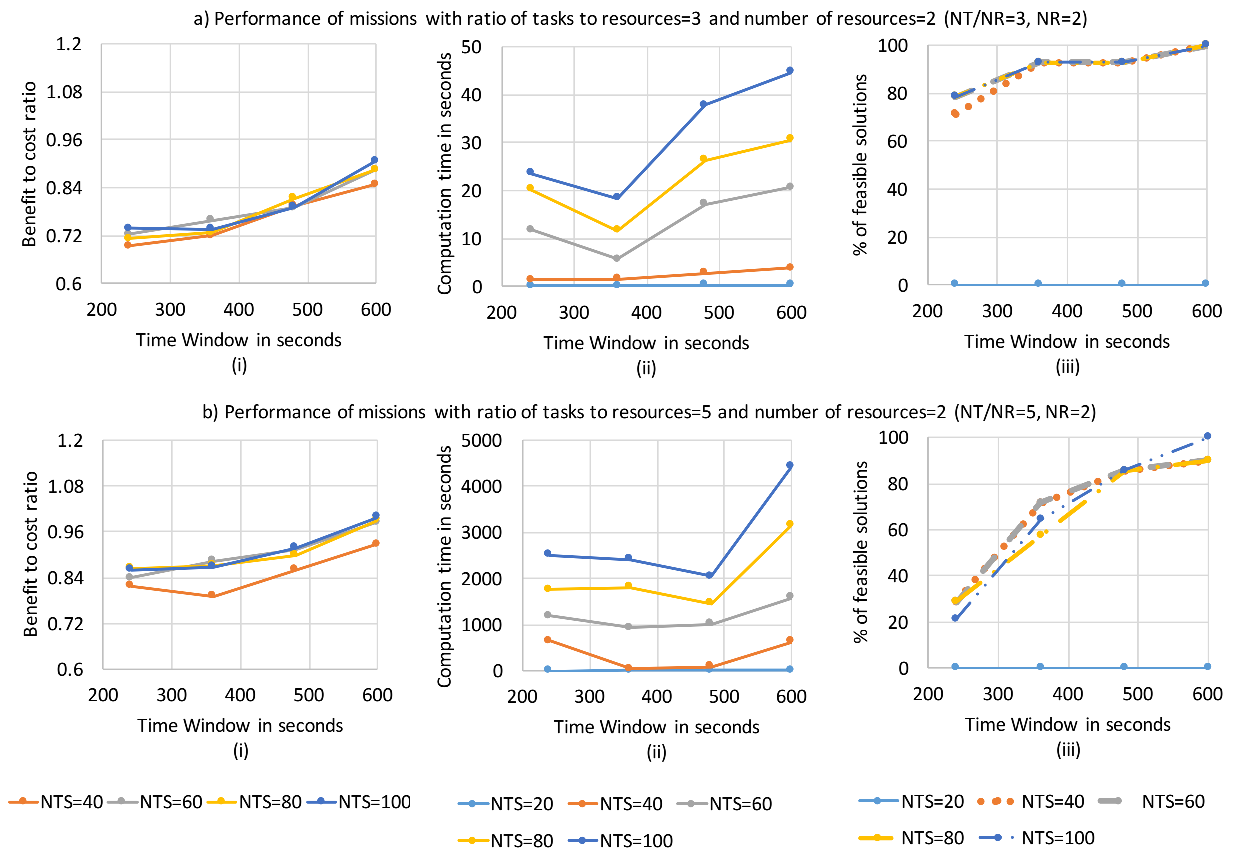

Fig. 11: Effect of varying time window (TW) size and number of time slices (NTS) on mission performance metrics for two vehicle resources (NR=2).

For the cases where the number of time slices was 20 (NTS=20) no feasible solutions were generated and these cases are not included in the benefit to cost ratio graphs (a) and (d) in Fig. 11 and Fig. 12. Additional analysis revealed that 90% of the cases with NTS=60 were physically feasible but failed due to insufficient time resolution for pre task sensor installation. This part of the study validates the formulation of the sensor selection constraints (section IV). Where there is a mission failure due to sensor installation constraints, the operator may decide to relax the sensor requirements and obtain a very quick response with low time resolution. From the results in Fig. 11 and Fig. 12 we can deduce that with larger time windows and higher time resolutions the quality of the solution, measured by the benefit to cost ratio, increases. The increase in the solution quality comes at the expense of computation time. Increasing the time resolution does not always lead to a better solution, it does however lead to more computation time. Reduction of the time resolution and time window size both reduce the percentage of feasible solutions (Fig. 11aiii, Fig. 11biii, Fig.12aiii and Fig. 12biii). The most significant effect is from reduction of time window size. The larger the task to resources ratio ($\frac{\mathrm{NT}}{\mathrm{NR}}$) the more time resolution is required to obtain a good solution. With a large enough time window most missions can be solved.

In the case of $\frac{\mathrm{NT}}{\mathrm{NR}} = 3$ (Fig. 11a and Fig. 12a) the best option that gives 100% feasible solutions with low computation time and reasonable benefit to cost ratio for both NR=2 and NR=3 is at NTS=40 and TW=600. The smallest time window that results in a reasonable solution quality at a low computational cost is 360 seconds. This allows 93% of problems to be solved with NR=2 and 100% with NR=3.

In the case of $\frac{\mathrm{NT}}{\mathrm{NR}} = 5$ (Fig. 11b and Fig. 12b) there are more tasks to be allocated hence there are more variables in the allocation model. There is a big jump in the computation time from $\frac{\mathrm{NT}}{\mathrm{NR}} = 3$. The operator may decide to sacrifice the quality of the solution to achieve a good computation time by taking NTS=40 and TW=600 which can perform 85% of the missions with NR=2 and 100% with NR=3. However, if the operator is interested in a high solution quality, computation time can be traded with benefit to cost ratio by taking NTS=60 and TW=600. This will result in the best benefit to cost ratio with the lowest computation time, solving 85% of the problems with NR=2 and 100% with NR=3.

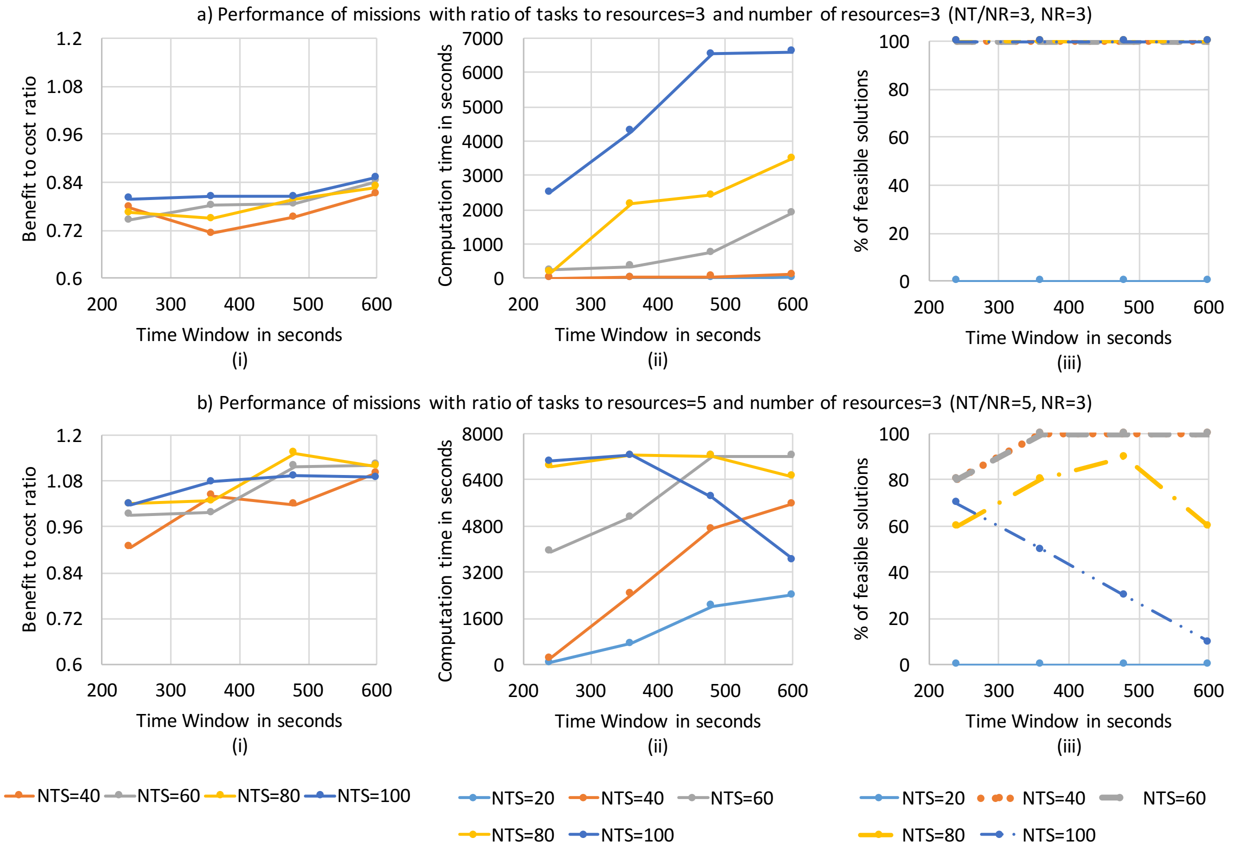

Fig. 12: Effect of varying time window (TW) size and number of time slices (NTS) on mission performance metrics for three vehicle resources (NR=3).

In general, the cases where the solutions are infeasible are the ones where time windows for multiple tasks are overlapping, requiring limited resources to perform many tasks in a very short period of time. Another class of infeasible solutions is due to vehicle resources having insufficient time to land and change their sensors. Increasing the time window size may eventually produce a feasible solution, but at a computation cost. For the purposes of this study, the computation time limit for the simulation of each mission was set at 7200 sec (2hrs). Some larger problems were identified as unfeasible due to computation time exceeding this limit (Fig. 11biii). Some of these missions are feasible with a smaller NTS. This is seen with NTS=80 and NTS=100 at TW=600 (Fig. 11biii). In Fig. 12bii and Fig. 12biii, we notice that for NTS=60, 80 and 100 the trends are counter intuitive. The percentage of feasible solutions dropped while the average computational time increased for NTS=60. This is due to reaching the 2 hrs time limit before finding a feasible solution for some complex missions. The percentage of feasible solutions dropped together with a large drop in the average computation time for NTS=80 and 100. This is due to missions being sufficiently large that there was insufficient memory in the simulation computer to store the variables before starting the simulation. The memory issue was detected within few seconds hence the simulation time is recorded as being very short.

Many factors affect the choice of an appropriate time window such as resource endurance, cruising speed and mission complexity. The system operator has to consider the type of missions expected and find a reasonable minimum time window that allows the resources to perform all tasks within their required time windows. Based on the simulation results summarized in Fig. 11 and Fig. 12, we can estimate a minimum time window value using the following heuristic equation. $\mathrm{TW_{min}} = \frac{1}{3} \times \frac{\mathrm{NT}}{\mathrm{NR}} \times (\text{Longest Travel Time} + \text{Average Execution Time})$

The Longest Travel Time is the travel time taken by the slowest resource to travel between the most distant two points in the operation area. In the cases considered, the longest travel time is 258 seconds, and the longest execution time is 60 seconds. The minimum time window recommended is thus 318 seconds for $\frac{\text{NT}}{\text{NR}} = 3$ and 530 seconds for $\frac{\text{NT}}{\text{NR}} = 5$.

The results presented in Fig. 5-10 for the baseline scenario ($\frac{\text{NT}}{\text{NR}} = 2.25$, NR=4, NTS=25 and a maximum TW=900 seconds) can be compared with the results presented in Fig. 11 and Fig. 12 for the performance evaluation scenario sets. The evaluation sets considered more complex cases ($\frac{\text{NT}}{\text{NR}} = 3$, NR=3, NTS=40 and TW=360 seconds) that were computed with an average computation time of 12.64 seconds which is similar to the results obtained for the baseline scenario.

3. Comparison of the effect of limiting the computation time to 120 seconds

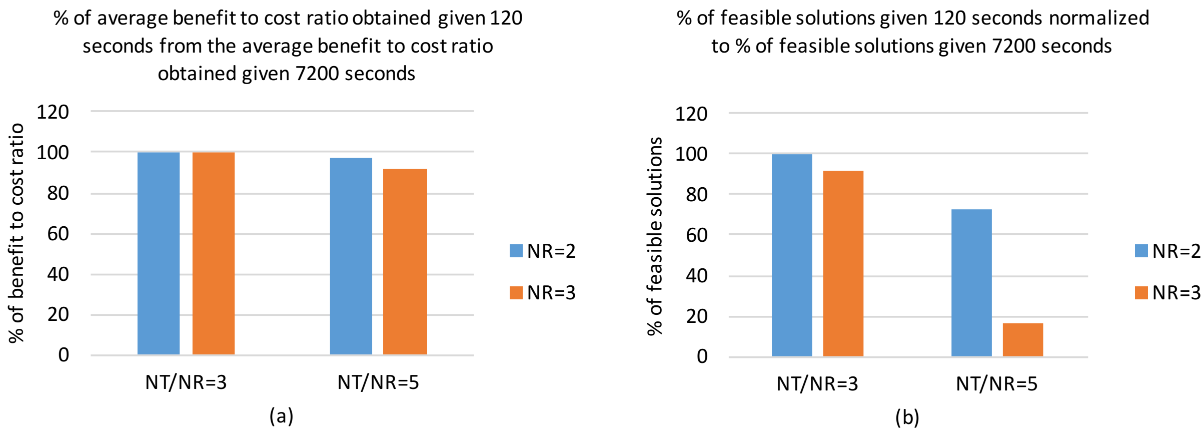

Fig. 13: Results of limiting the CPLEX computation time to 120 seconds.

The simulations conducted as part of the study of TW and NTS effects on the performance scenarios were repeated with a time limit of 120 seconds (2 min) instead of 7200 seconds (2 hrs). The average benefit to cost ratio of the solutions that are obtained with 120 seconds time limits are compared to their counterpart under the 7200 seconds time limit, Fig. 13a. The results in Fig. 13 show that in the case $\frac{\text{NT}}{\text{NR}} = 3$ the average benefit to cost ratio is reduced by 0.3% for NR=2 and by 0% for NR=3. For the more complex cases with $\frac{\text{NT}}{\text{NR}} = 5$ the average benefit to cost ratio dropped by 3.1% for NR=2 and by 8.5% for NR=3. The quality of the solution seems to remain very similar with the change of time limits; however, the time limit will prevent the solver from finding feasible solutions to feasible problems that simply require more computational time. This is evident in the large drop of percentage of feasible solutions with $\frac{\text{NT}}{\text{NR}} = 5$:73% for NR=2, 15% for NR=3. For the less complex problems with $\frac{\text{NT}}{\text{NR}} = 3$, the percentage of feasible solutions obtained in 120 seconds computational window was 99.5% for NR=2 and 91.6% for NR=3. This is still a decent percentage considering the size of the problems and their complexities. We can deduce that for medium size problems, limiting the CPLEX computation time to 120 seconds will yield very good results however for larger problems this limitation is too constraining.

4. Comparison of allocation methods

The literature is full of task allocation methods that tackle different subsets or combinations of problems; however, we were unable to find heuristics or other allocation methods that take all the constraints that we presented into account. In light of this, we compare the performance of the ILP model with a well-known distributed system instead, the Consensus Based Bundle Algorithm (CBBA) developed by Choi et al in [28]. We also developed a greedy algorithm that simply allocates the task that provides the highest reward in the shortest amount of time to the available resources. The greedy algorithm is there to show the comparison to a simplistic approach. Three sets with 10 random missions each were generated and solved using three distinct allocation methods. The results of the 90 runs are shown in Fig. 14.

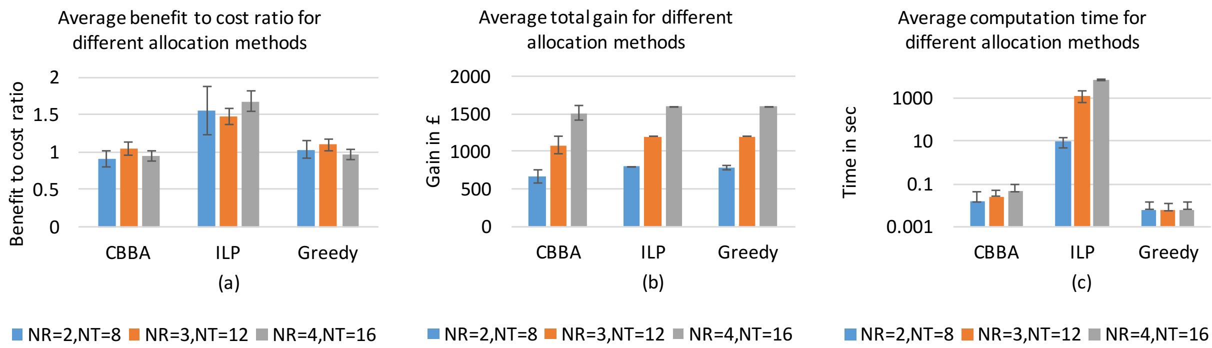

Fig. 14: Comparison of performance of ILP,CBBA and Greedy allocation methods.

Three performance metrics were used for the comparison: average benefit to cost ratio over the 10 missions in each set, the average of the total gain obtained and the average computation time. The average total gain also shows the number of tasks that were allocated since the reward for all tasks is equal to £100. Figure 14 (a) shows that the ILP model provides a solution that is 50% better than CBBA and 40% better than the greedy algorithm. Figure 14 (b) shows that in some cases, CBBA and the Greedy algorithm cannot allocate all the required tasks however the ILP model was able to find a feasible allocation for all the tasks. The main drawback of the ILP model comes from the computation time which is an expected issue. The CPLEX solver we are using is an optimal solver that requires a considerable amount of time to produce an optimal solution. Unlike the CBBA or the Greedy algorithms that took milliseconds to find an allocation, the CPLEX solver required minutes to find the optimal solution. The quality of the solution and the high benefit to cost ratio delivered comes at increased computational cost. This is mitigated against the novel features offered by the ILP model that enable an effective and efficient universal multi UAV resource allocation system.

Note that the ILP model developed in this paper can be used with a commercial solver like CPLEX to solve medium size problems with $\frac{NT}{NR} = 3$, NR=[2, 3], NTS=60 and TW=360 within 30 seconds. In a civilian multi UAV sharing system, 30 seconds is an acceptable computation time. The operator can also make use of the 120 seconds computation time limit and increase the time resolution of the solution or the time window allowance. CPLEX is not the only solver for ILP models, and larger problems will require faster solving methods. The ILP model developed is a stepping stone in the development of a quick holistic dynamic resource allocation system. The use of commercial solvers like CPLEX is to study the behavior and performance of the allocation model without focusing too much on the cost of computation time.

VI. Conclusions

A novel ILP formulation for autonomous vehicle resource allocation has been developed and validated that is able to produce an optimal resource schedule, within the set limitations, exploiting the ability of the resources to land, refuel and change their sensor suite in response to realistic multiple vehicles service provision scenarios. The simulation scenarios demonstrated the novel capabilities of the model to dynamically select sensor options, transport cargo with transshipment, choose *Intersection* tasks, plan for operational logistics including take off, landing and refueling at multiple bases. In addition, the simulations showed the adaptability of the model to different operational goals and its accommodation to interdependencies between heterogeneous tasks.

The presented model can be re-solved dynamically to accommodate new requests during execution. The ability to trade the resolution of the solution with computation time allows the user to adjust the time resolution of the solution based on real-time needs. The implementation can be extended to include additional environmental or operational constraints that may arise in the future. The effect of the coefficients in the cost function was prominent, particularly the coefficient of importance of finishing time and completion time. The more importance given to the completion time, the greater the trade with operational cost. Over-constraining a problem may render it unfeasible very easily especially when using narrow time windows. The implementation of this ILP model in a real time scenario requires the development of a faster solver that allows the system to produce a timely dynamic response.

The current model is designed to work with deterministic mission scenarios where all the inputs are certain. Future work must consider cases where there is uncertainty in the input data and how the model can be robust against those uncertainties. Uncertainty can be accommodated by allowing the model to find feasible solutions using a subset of the requested tasks when the full set of tasks is infeasible.

References

[1] T. Shima and S. Rasmussen, Eds., *UAV Cooperative Decision and Control*. Society for Industrial and Applied Mathematics, 2009.

[2] C. C. Murray and M. H. Karwan, "An extensible modeling framework for dynamic reassignment and rerouting in cooperative airborne operations," *Naval Research Logistics (NRL)*, vol. 57, no. 7, pp. 634–652, Aug. 2010.

[3] F. Mufalli, R. Batta, and R. Nagi, "Simultaneous sensor selection and routing of unmanned aerial vehicles for complex mission plans," *Computers & Operations Research*, vol. 39, no. 11, pp. 2787–2799, Nov. 2012.

[4] D. Turra, L. Pollini, and M. Innocenti, "Fast unmanned vehicles task allocation with moving targets," in *2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601)*, 2004, p. 4280–4285 Vol.4.

[5] E. Frazzoli, J. Enright, M. Pavone, and K. Savla, "UAV Routing and Coordination in Stochastic, Dynamic Environments," in *Handbook of Unmanned Aerial Vehicles*, K. P. Valavanis and G. J. Vachtsevanos, Eds. Dordrecht: Springer Netherlands, 2015, pp. 2079–2109.

[6] R. A. Wise and R. T. Rysdyk, "UAV coordination for autonomous target tracking," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, pp. 3210–3231, 2006.

[7] G. Varela, P. Caamamo, F. Orjales, A. Deibe, F. Lopez-Pena, and R. J. Duro, "Swarm intelligence based approach for real time UAV team coordination in search operations," *2011 Third World Congress on Nature and Biologically Inspired Computing*, pp. 365–370, Oct. 2011.

[8] A. Raptopoulos, D. Damm, P. Santana, M. Ling, and I. Baruchin, "Transportation using network of unmanned aerial vehicles." US Patents number US20140032034, 2014.

[9] M. Bernard, K. Kondak, I. Maza, and A. Ollero, "Autonomous transportation and deployment with aerial robots for search and rescue missions," *Journal of Field Robotics*, vol. 28, no. 6, pp. 914–931, 2011.

[10] D. Aume, Nilss M ; Mills, Robert G ; Gillio, Aldo A ; Sebasky, Gene ; Wartluft, "Summary Report of AMRL Remotely Piloted Vehicle (RPV) System Simulation Study V Results.," 1977.

[11] I. Maza, A. Ollero, E. Casado, and D. Scarlatti, "Classification of Multi-UAV Architectures," in *Handbook of Unmanned Aerial Vehicles*, Dordrecht: Springer Netherlands, 2015, pp. 953–975.

[12] S. S. Ponda, L. B. Johnson, A. Geramifard, and J. P. How, "Cooperative Mission Planning for Multi-UAV Teams," in *Handbook of Unmanned Aerial Vehicles*, K. P. Valavanis and G. J. Vachtsevanos, Eds. Dordrecht: Springer Netherlands, 2015, pp. 1447–1490.

[13] J. Acevedo, B. Arrue, I. Maza, and A. Ollero, "Distributed Approach for Coverage and Patrolling Missions with a Team of Heterogeneous Aerial Robots Under Communication Constraints," *International Journal of Advanced Robotic Systems*, p. 1, 2013.

[14] J. Kim and J. R. Morrison, "On the Concerted Design and Scheduling of Multiple Resources for Persistent UAV Operations," *Journal of Intelligent & Robotic Systems*, vol. 74, no. 1–2, pp. 479–498, Apr. 2014.

[15] F. M. Delle Fave, A. Rogers, Z. Xu, S. Sukkarieh, and N. R. Jennings, "Deploying the max-sum algorithm for decentralised coordination and task allocation of unmanned aerial vehicles for live aerial imagery collection," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 469–476.

[16] C. E. Pippin, H. Christensen, and L. Weiss, "Dynamic, cooperative multi-robot patrolling with a team of UAVs," *SPIE*, vol. 8741, pp. 874103-874103–8, May 2013.

[17] C. Rasche, C. Stern, W. Richert, L. Kleinjohann, and B. Kleinjohann, "Combining Autonomous Exploration, Goal-Oriented Coordination and Task Allocation in Multi-UAV Scenarios," in *2010 Sixth International Conference on Autonomic and Autonomous Systems*, 2010, pp. 52–57.

[18] E. Yanmaz, R. Kuschnig, M. Quaritsch, C. Bettstetter, and B. Rinner, "On path planning strategies for networked unmanned aerial vehicles," in *2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2011, pp. 212–216.

[19] J. Han, Y. Xu, L. Di, and Y. Chen, "Low-cost multi-UAV technologies for contour mapping of nuclear radiation field," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 70, pp. 401–410, 2013.

[20] M. Alighanbari, "Task assignment algorithms for teams of UAVs in dynamic environments," M.S. Thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, 2004.

[21] S. Moon, D. H. Shim, and E. Oh, “Cooperative Task Assignment and Path Planning for Multiple UAVs,” in *Handbook of Unmanned Aerial Vehicles*, Dordrecht: Springer Netherlands, 2015, pp. 1547–1576.

[22] M. L. Cummings, “Operator Interaction with Centralized Versus Decentralized UAV Architectures,” in *Handbook of Unmanned Aerial Vehicles*, Dordrecht: Springer Netherlands, 2015, pp. 977–992.

[23] M. Alighanbari and J. P. How, “Decentralized Task Assignment for Unmanned Aerial Vehicles,” in *Proceedings of the 44th IEEE Conference on Decision and Control*, 2005, pp. 5668–5673.

[24] D. Dionne and C. a. Rabbath, “Multi-UAV Decentralized Task Allocation with Intermittent Communications: the DTC algorithm,” in *2007 American Control Conference*, 2007, pp. 5406–5411.

[25] P. B. Sujit and J. B. Sousa, “Multi-UAV task allocation with communication faults,” in *2012 American Control Conference (ACC)*, 2012, pp. 3724–3729.

[26] M. Pujol-Gonzalez, J. Cerquides, P. Meseguer, J. A. Rodríguez-Aguilar, and M. Tambe, “Engineering the Decentralized Coordination of UAVs with Limited Communication Range,” in *The Conference of the Spanish Association for Artificial Intelligence (CAEPIA 2013) MADRID: Springer-Verlag*, 2013, pp. 199–208.

[27] C. C. Murray, “Dynamic reassignment and rerouting in cooperative airborne operations,” The State University of New York at Buffalo, 2010.

[28] H. Choi, L. Brunet, and J. How, “Consensus-based decentralized auctions for robust task allocation,” *Robotics, IEEE Transactions on*, vol. 25, no. 4, pp. 912–926, 2009.

[29] Han-Lim Choi, A. K. Whitten, and J. P. How, “Decentralized task allocation for heterogeneous teams with cooperation constraints,” in *Proceedings of the 2010 American Control Conference*, 2010, pp. 3057–3062.

[30] A. Richards and J. P. How, “Aircraft trajectory planning with collision avoidance using mixed integer linear programming,” in *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, 2002, vol. 3, pp. 1936–1941.

[31] A. Richards, J. Bellingham, M. Tillerson, and J. How, “Coordination and Control of Multiple UAVs,” in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2002, no. August, pp. 1–11.

[32] S. Karaman, E. Koyuncu, and G. Inalhan, “Innovative Collaborative Task Allocation for UAVs,” in *Handbook of Unmanned Aerial Vehicles*, K. P. Valavanis and G. J. Vachtsevanos, Eds. Dordrecht: Springer Netherlands, 2015, pp. 1601–1617.

[33] a. Ruangwiset, “Path generation for ground target tracking of airplane-typed UAV,” *2008 IEEE International Conference on Robotics and Biomimetics*, pp. 1354–1358, Feb. 2009.

[34] M. Kothari, I. Postlethwaite, and D.-W. Gu, “Multi-UAV path planning in obstacle rich environments using Rapidly-exploring Random Trees,” in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, 2009, pp. 3069–3074.

[35] M. Faied, A. Mostafa, and A. Girard, “Vehicle Routing Problem Instances : Application to Multi-UAV Mission Planning,” *AIAA Guidance, Navigation and Control Conference*, no. August, 2010.

[36] B. Bethke, M. Valenti, and J. How, “UAV Task Assignment,” *IEEE Robotics & Automation Magazine*, vol. 15, no. 1, pp. 39–44, Mar. 2008.

[37] N. Nigam, S. Bieniawski, I. Kroo, and J. Vian, “Control of multiple UAVs for persistent surveillance: Algorithm and flight test results,” *IEEE Transactions on Control Systems Technology*, vol. 20, no. 5, pp. 1236–1251, 2012.

[38] B. D. Song, J. Kim, and J. R. Morrison, “Towards real time scheduling for persistent UAV service: A rolling horizon MILP approach, RHTA and the STAH heuristic,” in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2014, pp. 506–515.

[39] C. Murray and M. Karwan, “A branch-and-bound-based solution approach for dynamic rerouting of airborne platforms,” *Naval Research Logistics*, vol. 60, no. 2, pp. 141–159, 2013.

[40] S. Karaman and E. Frazzoli, “Linear temporal logic vehicle routing with applications to multi-UAV mission planning,” *International Journal of Robust and Nonlinear Control*, vol. 21, no. May, pp. 1372–1395, 2011.

[41] N. Michael, J. Fink, and V. Kumar, “Cooperative manipulation and transportation with aerial robots,” *Autonomous Robots*, vol. 30, no. 1, pp. 73–86, Jan. 2011.

[42] I. Maza, F. Caballero, J. Capitán, J. R. Martínez-de-Dios, and A. Ollero, “Experimental Results in Multi-UAV

Coordination for Disaster Management and Civil Security Applications," *Journal of Intelligent & Robotic Systems*, vol. 61, no. 1–4, pp. 563–585, Dec. 2010.

[43] E. Benavent and A. Martínez, "Multi-depot Multiple TSP: a polyhedral study and computational results," *Annals of Operations Research*, vol. 207, no. 1, pp. 7–25, Nov. 2011.

[44] C. Sabo, "SMART Heuristic for Pickup and Delivery Problem (PDP) with Cooperative UAVs," *AIAA InfoTech*, no. March, pp. 1–19, 2011.

[45] A. C. Chapman, R. A. Micillo, R. Kota, and N. R. Jennings, "Decentralized Dynamic Task Allocation Using Overlapping Potential Games," *The Computer Journal*, vol. 53, no. 9, pp. 1462–1477, Mar. 2010.

[46] M. Alighanbari and J. How, "Robust decentralized task assignment for cooperative UAVs," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, no. August, 2006.

[47] S. Erdoğan and E. Miller-Hooks, "A Green Vehicle Routing Problem," *Transportation Research Part E: Logistics and Transportation Review*, vol. 48, no. 1, pp. 100–114, Jan. 2012.

[48] P. Toth and D. Vigo, *The Vehicle Routing Problem*, vol. 9. Philadelphia: Society for Industrial and Applied Mathematics, 2002.

[49] D. Davendra, *Traveling Salesman Problem, Theory and Applications*. Rijeka, Croatia: InTech, 2010.

[50] IBM, "IBM ILOG CPLEX Optimization Studio V12.6.1 documentation." [Online]. Available: http://www.ibm.com/support/knowledgecenter/#!/SSSA5P_12.6.1/ilog.odms.studio.help/Optimization_Studio/topics/COS_home.html. [Accessed: 16-May-2016].