

Asset Allocation and Routing, with Games and Repair, under Uncertainty

Nikolaos Papadakos, Berç Rustem and George Tzallas-Regas
Imperial College London, 180 Queen's Gate, London, SW7 2AZ

Abstract

In this paper we are concerned with task assignment and routing under uncertainty. We present our advances in this area since the last SEAS DTC conference in Edinburgh [10,11]. During the last year, we included the vehicle assignment and routing, of multiple assets on the same opponent. Additionally we consider implications of game theory, where we need to optimise against the opponents' optimal strategy. Finally, we consider repairing solutions once something unexpected has happened in our initial plan.

1. Software Development for Multi-asset Task Assignment by Maximising Target Score

In this section we are concerned with task assignment models, where a number of assets are allocated to some targets, in order to complete a certain task. One way round the assignment problem is to give each target a score, and to do the assignment in a way that maximises the target score. Part of the constraints of course models whether an asset, if assigned, can actually contribute to the success of the task through a reachability matrix mainly. The advantage of such a model is that it is linear and can be solved efficiently using commercial applications like CPLEX [1]. In order to allow assets to collaborate though researchers have resorted to models that try to maximise the probability of success of the task. Such models, though, are nonlinear in the objective function, and different solvers have to be used which may not be as efficient. In this work, we propose a model that achieves collaboration between the assets and the objective function of the model is still linear. In order to articulate this model, the objective function uses target score information, but we also use information about the nature of the assets and the targets, which require no more than what would be required to express constraints of the linear model.

1.1 Introduction

We are interested in task allocation models, in which we seek to assign a set of V assets to a set of T tasks in an optimal way. Our concern lies mainly with models in which optimality is quantified using target scores. That is to say, we give each task a score, and if the task can be carried out completely, we contribute to the assignment the score of the target. The mission of the model is to maximise the target score and to model capacity constraints of both the assets and the tasks.

1.2 Multi-asset Assignment Model

We assume that each asset consists of a number of sub-assets, that describe the ability of the asset to contribute to the task, and that could be used, in traditional models, to formulate a reachability matrix for the assignment problem. For each target, we describe the number of sub-assets that are needed to completely acquire the maximum scored assigned to the target. Let us examine the following example:

1.2.1 Example

Assume for example that the problem at hand is to load a number of targets to our assets. If we know the dimension of the targets (might be different), the sub-assets of each asset could be defined as the

volume each asset can carry. It is obvious, that voluminous targets will be excluded by the use of a reachability matrix which is formed using this information.

In our approach though, we dispense with reach ability matrices, and use the volume information in order to try to get two or more assets to jointly lift the voluminous target.

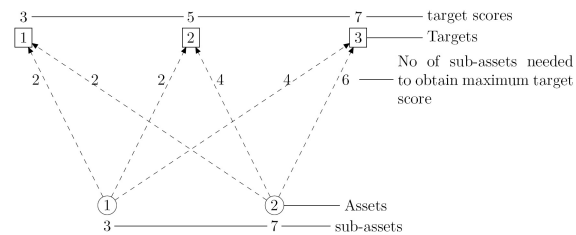


Figure 1: Task assignment example

As is seen in the example:

- Target 1 is reachable by both assets,
- Target 2 is reachable by both assets,
- Target 3 is reachable by asset 2 only.

With a model that uses a reach ability matrix, the maximum score would be 12, which is achieved by assigning:

- asset 1 to target 2, with 2 sub-assets. Asset 1 will contribute a score of 5 to the total, and will have one sub-asset remaining.
- asset 2 to target 3. Asset 2 will contribute a score of 7 to the total, and will have on sub-asset remaining.

We could say that we have achieved 12/15=80% of the mission, since target 1 has not been assigned. The sub-assets that have not been used, could in fact be used collectively as follows:

- Asset 1, needs to have 2 sub-assets to attain the maximum score of 3, if assigned to target 1. With 1 sub-asset it would contribute (1/2) of the total target score, that is 1.5.
- Asset 2, needs to have 2 sub-assets to attain the maximum score of 3, if assigned to target 1. With 1 sub-asset it

would contribute (1/2) of the total target score, that is 1.5.

Therefore, if the assets are allowed to collaborate, then a total of 15 would be attained, which is the maximum possible score. In this case, we could say that 100% of the mission has been accomplished.

In order to generate such an assignment, we need to take into account the number of sub-assets that have to be assigned to each target, and discount the objective function of the original model accordingly. The following notation will be used in the model:

T : the set of targets

V : the set of assets / vehicles

then the variables of the model are (i) x_{vt} , which is a binary variable set to 1 if $v \in V$ is assigned to $t \in T$, 0 otherwise (ii) m_{vt} is a non-negative integer for the number of sub-assets of $v \in V$ assigned to $t \in T$.

The parameters¹ of the model are (i) s_t is the score of target $t \in T$, (ii) vm_v is the number of sub-assets of $v \in V$ and (iii) tm_{tv} is the number of sub-assets of $v \in V$ that we need to assign to $t \in T$ in order to attain 100% of s_t . The model for this problem is²

$$\begin{aligned} \max \quad & \sum_{v \in V, t \in T} \frac{m_{vt} s_t}{tm_{tv}} \\ \text{s.t.} \quad & \sum_{t \in T} m_{vt} \leq vm_v, \\ & \sum_{v \in V} \frac{s_t m_{vt}}{tm_{tv}} \leq s_t, \\ & x_{vt} \leq m_{vt}, \\ & m_{vt} \geq x_{vt} \min\{vm_v, tm_{tv}\} \\ & x_{vt} \in \{0, 1\}, \\ & m_{vt} \in N^+ \end{aligned}$$

where N^+ is the set of non-negative integers. As stated earlier, x_{vt} are binary

¹ a minimum of them, at least

² for space reasons, after the constraints omit $v \in V, t \in T$, where appropriate

variables that state which asset is assigned to which target. Variable m_{vt} is a non-negative integer that counts the number of sub-assets of asset v that are assigned to target t . Obviously, if an asset is assigned to a target, it has to be assigned with a minimum of a sub-asset, as constraint

$$x_{vt} \leq m_{vt}, \quad v \in V, t \in T,$$

reveals. In addition the number of sub-assets of asset v assigned to a target t - if assigned - cannot exceed either the available (vm_v) ones of asset v , or the ones required to achieve 100% of the score of the target t . Therefore we impose the constraint

$$\begin{aligned} m_{vt} &\leq x_{vt}vm_v, & v \in V, t \in T \\ m_{vt} &\leq x_{vt}tm_{tv}, & v \in V, t \in T \end{aligned}$$

which has been condensed to $m_{vt} \leq x_{vt} \min\{vm_v, tm_{tv}\}$. Furthermore, an asset v can visit as many targets as its capacity allows, which is captured using:

$$\sum_{t \in T} m_{vt} \leq vm_v, \quad v \in V$$

The traditional way to model the objective function would be

$$f(x) = \sum_{v \in V, t \in T} x_{vt} s_t$$

In this objective function, we implicitly assume that if an asset v is assigned to target t , it achieves 100% of the score. In the objective function of the model

$$f(x, m) = \sum_{v \in V, t \in T} \frac{m_{vt} s_t}{tm_{tv}}$$

we are again trying to accumulate the highest possible score for the mission, but this depends on the number of sub-assets of asset v that we allocate to target t .

1.2.2 Example Results

If we were to use this model against the aforesaid example, we would obtain the following results:

Asset	Target	Sub-assets	Completion	Score
1	2	1	50%	2.5
1	3	2	50%	3.5
2	1	1	50%	1.5
2	2	2	50%	2.5
2	3	3	50%	3.5
Total score				15

We observe that the final score reveals that the mission has been completed 100%, and it is something that is readily available. Therefore, the advantage of such a formulation of the target score model is two-fold:

1. It allows the assets to collaborate and, possibly, achieve higher value of the target score.
2. It allows the decision maker to draw quantitative remarks about the completion of the mission.

1.3 Model Uncertainties

Uncertainties in the problem come through the problem data and emanate mainly from the fact that our observation of the target might be inaccurate. Therefore, the score of each target (s_t) is expected to lie within some lower and upper bounds. Similarly tm_{tv} and vm_v are expected to lie within similar lower and upper bounds. For example, if we use ξ to denote a random vector then $s_t(\xi)$ denotes the expected value of s_t under the probability distribution of ξ . Using similar notation for the rest of the uncertain variables, we can write the model as

$$\begin{aligned} \max \quad & \sum_{v \in V, t \in T} \frac{m_{vt} s_t(\xi)}{tm_{tv}(\xi)} \\ \text{s.t.} \quad & \sum_{t \in T} m_{vt} \leq vm_v(\xi) \\ & \sum_{v \in V} \frac{s_t(\xi) m_{vt}}{tm_{tv}(\xi)} \leq s_t(\xi), \\ & x_{vt} \leq m_{vt}(\xi), \\ & x_{vt} \min\{vm_v(\xi), tm_{tv}(\xi)\} \geq m_{vt}, \\ & x_{vt} \in \{0, 1\}, \\ & m_{vt} \in \mathbb{N}^+. \end{aligned}$$

1.4 Model Implementation

In the model implementation we try to maximise the pre-assigned target score, which is discounted by the number of missiles spent on each target. We also take into account fuel and missiles in our constraints and also try to perform some routing.

The model proceeds using origin-destination pairs. The current position of an asset is its origin and the target it is going visit its destination. Assets move along these pairs. The model has one objective and three sets of constraints. The first set of constraints deals with fuel, the second set of constraints deals with missiles and the third set of constraints forces assets to visit targets, starting and stopping at their depot.

In order to present the model the following sets are introduced:

1. \mathcal{V} : the set of vehicles
2. \mathcal{T} : the set of targets, which partially form the set of origins / destinations.
3. \mathcal{T}_0 : the set of targets extended to include the depot, the ultimate origin and destination of each vehicle

We will use n_X to denote cardinality of set \mathcal{X} . Lower case letters will denote set elements, i.e. $v \in \mathcal{V}$ is a *vehicle*, $o \in \mathcal{T}_0$ or $o \in \mathcal{T}$ an *origin* and $d \in \mathcal{T}_0$ or $d \in \mathcal{T}$ a *destination*. Having these in mind we introduce the following set of variables:

1. x : a binary variable. This is set to 1 if $v \in \mathcal{V}$ goes from $o \in \mathcal{T}_0$ to $d \in \mathcal{T}_0$. It is easy to see that there are $n_V \times n_{T_0}^2$ of these.
2. x_f : a non-negative integer variable. For each $v \in \mathcal{V}$, this variable measures the fuel spent visiting targets. There are obviously n_V of these.
3. x_m : a non-negative integer variable. For each $v \in \mathcal{V}$, this variable measures how many missiles are spent going from $o \in \mathcal{T}_0$ to $d \in \mathcal{T}$. There is no need to spend

any missiles when going back to the depot and there are $n_V \times n_{T_0} \times n_T$ of these.

We also need a set of parameters to describe our model. These are the following:

- s_d : the score of each $d \in \mathcal{T}_0$. For simplicity, a score of 0 (zero) is assigned to the depot; there are n_{T_0} of these parameters.
- f_v : the fuel of each vehicle; $v \in \mathcal{V}$; there are n_V of these parameters.
- f_{vod} : the fuel that $v \in \mathcal{V}$ needs to go from $o \in \mathcal{T}_0$ to $d \in \mathcal{T}_0$; $n_V \times n_{T_0}^2$ of these parameters are needed in the model.
- m_v : the number of missiles that each $v \in \text{set } \mathcal{V}$ has; n_V in total.
- m_{vod} : the number of missiles that each $v \in \mathcal{V}$ has to spend when going from $o \in \mathcal{T}_0$ to $d \in \mathcal{T}_0$ in order to acquire 100% of the score of the destination; $n_V \times n_{T_0}^2$ in total.

Note that for the fuel the same letter is used for both the available and the necessary fuel. The number of subscripts is different though. One subscript is taken to mean the available fuel of the subscripted vehicle; three subscripts are used to denote the necessary fuel that the subscripted vehicle (first subscript) needs to go from the subscripted origin (second subscript) to the subscripted destination (third subscript). A similar comment applies for the missiles.

Having described the variables and the parameters of our model we can proceed to describe the objective and the constraints of our model. The idea is to visit targets, trying to acquire the maximum possible target score (a value between 0 and $\sum_{d \in \mathcal{T}} s_d$), respecting fuel and missiles.

1.4.1 Fuel Constraints

For fuel considerations we need two sets of constraints. One to measure the amount of fuel that vehicle v spends from origin v to destination d

$$x_v^f = \sum_{o \in \mathcal{T}_0, d \in \mathcal{T}_0} x_{vod} \cdot f_{vod}, \quad \text{for all } v \in \mathcal{V}, \quad (1a)$$

and one that states that no vehicle v can spend more fuel than it already has

$$x_v^f \leq f_v, \quad \text{for all } v \in \mathcal{V}. \quad (1b)$$

The first set of constraints has n_V variables and so does the second set.

1.4.2 Missile Constraints

As far as the missiles are concerned the following points have to be taken into account: A vehicle should not spend more missiles than it already has, or more than necessary to achieve 100% of the destination score. Mathematically for all $v \in \mathcal{V}, o \in \mathcal{T}_0, d \in \mathcal{T}$

$$x_{vod}^m \leq x_{vod} \cdot \min(m_v, m_{vod}). \quad (2a)$$

Note that the depot is not part of the asset destinations as far as the missiles are concerned, since there is no need to spend missiles when visiting the depot. The number of constraints in this set is $n_V \times n_{\mathcal{T}_0} \times n_{\mathcal{T}}$. We also add a set of constraints ($n_V \times n_{\mathcal{T}_0} \times n_{\mathcal{T}}$ in total) which state that, if you visit a target, then you have to spend at least a missile, that is for all $v \in \mathcal{V}, o \in \mathcal{T}_0, d \in \mathcal{T}$

$$x_{vod} \leq x_{vod}^m. \quad (2b)$$

In addition, no vehicle should go to more targets than it can handle, that is we also add the following (n_V in total) constraints

$$\sum_{o \in \mathcal{T}_0, d \in \mathcal{T}} x_{vod}^m \leq m_v, \quad \text{for all } v \in \mathcal{V}. \quad (2c)$$

Finally we add an extra set of constraints ($n_{\mathcal{T}}$) that are taken to mean that no vehicle should try to overachieve its partial mission, formulated as:

$$\sum_{v \in \mathcal{V}, o \in \mathcal{T}_0} \frac{x_{vod}^m}{m_{vod}} \leq 1, \quad \text{for all } d \in \mathcal{T}. \quad (2d)$$

Note here, that none of the constraints above contain an equality. This point will be discussed further when we describe the objective function of the model.

1.4.3 Time Constraints

Impose an upper bound on time:

$$x_{vo}^t \leq uw_o \sum_{d \in \mathcal{T}_0} x_{vod}, \quad (3a)$$

for all $v \in \mathcal{V}, o \in \mathcal{T}$.

Impose a lower bound on time:

$$x_{vo}^t \geq lw_o \sum_{d \in \mathcal{T}_0} x_{vod}, \quad (3b)$$

for all $v \in \mathcal{V}, o \in \mathcal{T}_0$.

Compute the time at which you need to be at d when departing from v

$$x_{vod}(x_{vo}^t + srv_o + trv_{vod} - x_{vd}^t) \leq 0 \quad (3c)$$

for all $v \in \mathcal{V}, o \in \mathcal{T}, d \in \mathcal{T}$. This constraint says that if you go to target d from origin o , then the time when you arrive at d (x_{vd}^t) should be greater than the time when you arrived at the origin, the time you spend servicing the origin and the time you need to travel from the origin to the destination ($x_{vo}^t + srv_o + trv_{vod}$). We note that in this constraint there is a variable (x_{vod}) multiplying another variable (x_{vo}^t, x_{vd}^t) Therefore this constraint is a non-linear constraint and such a model cannot be solved using LP technology. The constraint is actually a bilinear constraint (multiplication of linear terms), and can also be relaxed to yield a linear constraint. This is achieved by choosing a large number M_{od} and formulating a constraint as follows

$$(x_{vo}^t + srv_o + trv_{vod} - x_{vd}^t) \leq (1 - x_{vod})M_{od}, \quad (3d)$$

for all for all $v \in \mathcal{V}$, $o \in \mathcal{T}$, $d \in \mathcal{T}$. This constraint is indeed a linear constraint. This constraint can be interpreted as follows :

If we do arrive at destination d from origin o , then the time at which we arrive at d should be after the time we were at o including the time we need to spend servicing o and the time we need to travel from o to d . On the other hand, if we do not visit d , then the right hand side of the constraint is so big, that the constraint can be safely ignored.

1.4.4 Routing Constraints

In order to obtain some form of routing, we have to add a number of constraints that involve the binary variables x . We start by adding the following set of constraints for all $v \in \mathcal{V}$, $o \in \mathcal{T}$, $d \in \mathcal{T}$

$$\sum_{d_1 \in \mathcal{T}} x_{v0d_1} \geq x_{vod}, \quad (4a)$$

as well as

$$\sum_{o_1 \in \mathcal{T}} x_{vo_10} \geq x_{vod} \quad (4b)$$

for all $v \in \mathcal{V}$, $o \in \mathcal{T}$, $d \in \mathcal{T}$. The first set of constraints says that, if an asset starts visiting targets, it should do so starting from the depot. Similarly the second set of constraints ($n_V \times n_T^2$ in total) says that an asset, if it visits any targets, it should end at the depot. We also add the following n_V constraints

$$\sum_{t \in \mathcal{T}_0} x_{vtt} = 0, \quad \text{for all } v \in \mathcal{V} \quad (4c)$$

that does not allow vehicles to re-visit targets. In addition, we need to disallow assets from visiting nodes more than once, so we add the following $n_V \times n_{T_0}$ constraints for all $v \in \mathcal{V}$, $a \in \mathcal{T}_0$:

$$\sum_{b \in \mathcal{T}_0} x_{vab} + \sum_{b \in \mathcal{T}_0} x_{vba} \leq 2. \quad (4d)$$

The final two sets of constraints state that an asset should start and end at the depot doing a cycle ($n_V \times n_T$ in total) for all $v \in \mathcal{V}$, $b \in \mathcal{T}$

$$x_{v0b} \leq \sum_{c \in \mathcal{T}_0} x_{vbc} \quad (4e)$$

and that when performing a cycle, an asset should not go back to the same destination ($n_V \times n_T^2$ in total) for all $v \in \mathcal{V}$, $a \in \mathcal{T}$, $b \in \mathcal{T}$

$$x_{vab} \leq \sum_{c \in \mathcal{T}_0} x_{vbc} - x_{vba}. \quad (4f)$$

1.4.5 Objective Function

The objective of the model is to maximise the following term

$$\sum_{v \in \mathcal{V}, o \in \mathcal{T}_0, d \in \mathcal{T}} \frac{x_{vod}^m \cdot s_d}{m_{vod}}. \quad (5)$$

This means that for every missile that we spend on a target (going from o to d), we get a fraction $\frac{s_d}{m_{vod}}$ of its total target score. We will never get more than 100% as constraints (2d) stipulate. As already mentioned previously, the constraints on the number of missiles x_m involve inequalities only, given the model a freedom to choose the best it can do, within demanded and allowed limits.

2. Game Theoretic Treatment of the Problem

2.1 Expected Damage and Min-max

When solving the UV route planning problem so far it is assumed that the enemy has a fixed strategy that is known to us. However, is the opponent's strategy the best one possible? Is the opponent actually anticipating our next move? It is but fair to assume that the opponent desires to achieve the best possible strategy as we do too. This fact can be taken into account by game theory. Game theory [4] has been used for vehicle routing by [2,3,5,9,12].

2.2 Game Formulations

Let us assume that there are two participants, BLUE (B) and RED (R). We formulate the utility function as the difference of two payoffs.

$$P(w) = B(w) - R(w) \quad (6)$$

where $B(w)$ is the payoff for BLUE, $R(w)$ is the payoff for R and w is the vector of control variables. At every stage the strategy for each player is:

- BLUE: $\max P(w)$
- RED: $\max P(w)$

The payoff can take different forms, depending on which aspect of the game we wish to concentrate.

Reserve Maximisation: In this type of game we divide the number of assets into fighters and reserves. The rules of the game are as follows: (i) BLUE fighters can not only destroy RED fighters, but they can also destroy RED reserves; (ii) at every stage we sent a number of fighters to engage RED fighters and if possible destroy RED reserves. The invariants of the game (known to both players) are the probability of each fighter to destroy rival assets, the probability to survive every engagement and a number of replacements that every player can use at the beginning of every stage. The decision is how many of our assets should be used as fighters, in order to maximise reserves.

Group Effectiveness: In this type of game we consider group effectiveness. The idea is that the more assets we assign to a task, the higher the success of the task. At every stage of the game we need to decide how many of our assets we should assign to various tasks, bearing in mind that we only have a finite number of assets, and also bearing in mind that not all task have the same success probability. Each player is seeking to maximise the cumulative success probability of the tasks.

3. Software Development for Re-optimisation and Repair

During the implementation of a routing plan, one is often faced with dynamic changes due to the environment. For instance, a vehicle might be destroyed, might fail to accomplish a mission, or a new opponent might enter the field. Since the initial routing is under implementation, one has to respond to these changes by finding new solutions, reflecting the altered situation, as quickly as possible. There are two approaches for solving the Unmanned Vehicle Routing problem:

- by assuming that the opponent is following some reasonable predetermined strategy,
- by assuming that the opponent is dynamically refreshing his strategy with a goal to have the best possible result at the end.

3.1 Warm Starts

Depending on the type of the algorithm used to solve the problem, one may take advantage of warm starts. By warm starting an algorithm, we mean the restarting of the algorithm from a previously frozen stage. This is beneficial, especially if the data of the problem have not changed significantly. It is usually predicted that the solution will be in the vicinity of the solution of the frozen stage. Starting the algorithm from the beginning would obviously waste computational time. But exploiting warm starts is predicted to save a lot of time in locating the new optimum of the problem from a neighbouring point. Primal-Dual Interior point algorithm are easy to use and offer warm starts in a rather seamless manner (Gondzio and Grothey [6,7], Gondzio et al. [8]), although other algorithms might be of interest. The aforementioned references deal with warm starting PDIP algorithms for the linear case only (as part of a decomposition scheme mainly). We have not found sufficient

references for nonlinear problems, and that would be of great interest.

3.2 Constraint Programming

Constraint programming gives the opportunity to find such a quick solution using different clever search techniques. One additional advantage is that it naturally produces a routing that is a small perturbation of the initial one. That minimal discrepancy from the initial solution additionally eases the reimplementation of the routing, as a minimal number of changes will be required.

These methods start with a complete, possibly infeasible, assignment of values to variables and change the values of the variables until a feasible assignment is found. The criteria used to find feasible assignments are based on changing the variables that violate certain constraints, due to the effects of the environment. In effect one is repairing the violated constraints. In addition to maintaining feasibility one may endeavour to find a better solution by trying, at least, to find a local optimum to a given objective.

Repair methods are additionally used in solving problems which can be partitioned into quasi-independent simpler sub-problems. In our present problem a straightforward separation could be that between task assignment and vehicle routing. Further separations can be devised. In such cases it has to be underlined that solutions to the sub-problems which are useful for solving the complete problem may not be fully compatible with each other, or even completely feasible with respect to the full problem. It is up to the designer to decide which constraints are soft and which are weak. This is of course influencing the efficiency of the algorithms, and the user will have to consider the trade-offs.

Moreover, there are techniques such as conflict minimisation, which seek solutions that minimise infeasibility. These techniques can be treated as optimisation algorithms, whose constraints are wrapped into the optimisation function. However, they can also be treated as repair problems, which means that the constraints can propagate actively during problem solving.

Based on the above handling of constraints, it is possible to relax several constraints in a disjunctive form. Constraint programming can handle disjunctions contrary to mathematical programming.

Another possibility is the hybridisation of constraint programming techniques and mathematical programming. Such work has been ongoing for over a decade. These techniques typically involve the pruning of the search space in integer programming problems. Additionally, constraint programming serves in eliminating paths in resource constraint shortest path problems, typically used as sub-problems in column generation of scheduling problems.

4. Conclusions

We have presented the details of a task assignment and routing model. More than one asset is allowed to visit a target. Repair of an optimal strategy is also introduced. Repair is necessary when an unexpected event occurs and there may not be sufficient time to reinitiate an optimisation. Assignment is also considered from the point of view of game theory. Games are required to react to changes in dynamic and uncertain environments.

References

- [1] (2002). *ILOG CPLEX 8.0 User's Manual*. ILOG Inc, Mountain View, California.
- [2] Danskin, J. (1967). *The Theory of Max-Min and its Applications to Weapons Allocation Problems*. Springer-Verlag, New York.
- [3] Engevall, S., Göthe-Lundgren, M., and Värbrand, P. (2004). *The heterogeneous*

- vehicle routing game*. *Transportation Science*, 38(1):71–85.
- [4] Forgo, F., Szep, J., and Szidarovszky, F. (1999). *Introduction to the Theory of Games: Concepts, Methods, Applications*. Kluwer Academic, Dordrecht.
- [5] Ghose, D., Speyer, J. L., and Shamma, J. S. (2002). *A game theoretical multiple resource interaction approach to resource allocation in an air campaign*. *Annals of Operations Research*, 109(1):15–40.
- [6] Gondzio, J. and Grothey, A. (2003). *Reoptimization with the primal-dual interior point method*. *SIAM Journal on Optimization*, 13(3):842–864.
- [7] Gondzio, J. and Grothey, A. (2006). *A warm-start approach for large-scale stochastic linear programs*. Technical Report MS-06-004, School of Mathematics, University of Edinburgh, Edinburgh, UK.
- [8] Gondzio, J., Sarkissian, R., and Vial, J.-P. (1997). *Using an interior point method for the master problem in a decomposition approach*. *European Journal of Operational Research*, 101:577–587.
- [9] Hillestad, R. and Moore, L. (1996). *The theater-level campaign model: A research prototype for a new generation of combat analysis model*. Technical report, RAND Technical Report MR-388-AF/A, Rand Publication, Santa Monica, CA.
- [10] Papadakos, N., Rustem, B., and Tzallas-Regas, G. (2008a). *Integrated Task Assignment and Route Planning under Uncertainty*. In SEAS DTC, page B1, Edinburgh.
- [11] Papadakos, N., Rustem, B., and Tzallas-Regas, G. (2008b). *Sequential Updating of Decisions and Games/Plans under uncertainty*. In SEAS DTC, page B4, Edinburgh.
- [12] Vaughan, D., Kvitky, J., Henry, K., Gabriele, M., Park, G., Halverson, G., and Schweitzer, B. (1998). *Capturing the essential factors in reconnaissance and surveillance force sizing and mix*. Technical report, Project Air Force, RAND Documented Briefing, DB 199-AF, Rand Publication, Santa Monica, CA.

Acknowledgements

The work reported in this paper was funded by the Systems Engineering for Autonomous Systems (SEAS) Defence Technology Centre established by the UK Ministry of Defence. We acknowledge the support given by Jo Thoms in this research and Bob Clark for his expertise in

defining the domain. We also acknowledge the help by Duncan McCreddie when integrating our software package with the military system used by BAE Systems. Finally, we wish to thank participants in the SEAS DTC conference 2008 on Mission Planning and in particular David Oxenham for their insightful comments.