Department of Mathematics

Aircraft Mission Planning

Nils-Hassan Quttineh, Kristian Lundberg
Kaj Holmberg, Torbjörn Larsson

LiTH-MAT-R–2012/07–SE

Linköpings universitet

Linköpings universitet
581 83 Linköping

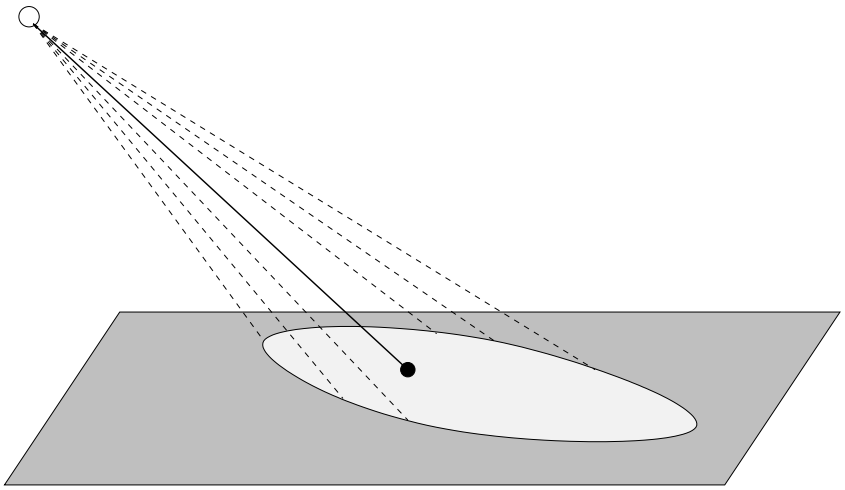
Linköping University
SE-581 83 Linköping, Sweden

Aircraft Mission Planning

Nils-Hassan Quttineh[a,1], **Kristian Lundberg**[a]
Kaj Holmberg[a], **Torbjörn Larsson**[a]

Linköping University

March 2012



[a] Department of Mathematics, Linköping University, SE-581 83 Linköping, Sweden
[1] Corresponding author. Tel: +46 13 282185. E-mail address: nils-hassan.quttineh@liu.se

This work is funded by the Swedish funding agency Vinnova.

Aircraft Mission Planning

Typeset by the authors in LaTeX2e documentation system.

Contents

Aircraft Mission Planning

Nils-Hassan Quttineh, Kristian Lundberg
Kaj Holmberg, Torbjörn Larsson

May 8, 2013

Abstract This paper deals with a Military Aircraft Mission Planning Problem, where the problem is to find time efficient flight paths for a given aircraft fleet that should attack a number of ground targets. Due to the nature of the attack, two aircraft need to rendezvous at the target, that is, they need to be synchronized in both space and time. At the attack, one aircraft is launching a guided weapon, while the other is illuminating the target. Each target is associated with multiple attack and illumination options. Further, there may be precedence constraints between targets, limiting the order of the attacks. The objective is to maximize the outcome of the entire attack, while also minimizing the mission time span. We present two mathematical models for this problem and compare their efficiency on some small test cases. We also provide some heuristic approaches since direct application of a general MIP solver to the mathematical model is only practical for smaller scenarios. The heuristics are compared and they successfully provide solutions to a number of scenarios.

Keywords: Aircraft Routing, Generalized Vehicle Routing Problem, Precedence, Synchronization, Military Operations Research, Decision Support.

1 Introduction

Military Mission planning is a complex task with a lot of interactions and prerequisites in both time and space. A mission plan shall be seen upon as a proposed sequence of actions that fulfills mission requirements from a higher hierarchical instance, and the smooth cooperation with own forces and other missions. Essential mission requirements are also provided by the rules of engagement, which in short refers to general permissions under the employment of military action.

As a consequence a mission plan holds the fundamental information for success and act as a 'map' in the phase of conduction. Objectives and constraints in the plan must be strictly met in order to synchronize and keep safety of own troops and equipment as well as a clear situation assessment throughout the mission. A lot of 'pieces' must work together like a clockwork and when they don't actions must be taken to coordinate and re-plan the mission or parts of the mission. As a high hierarchy commander, situation awareness is critical as well as fast and effective planning/replanning capability. He or she gets valid information about ongoing actions in time and space and when needed, replanning is ordered based on a best knowledge of what to do and when. So, solid intelligence and robust planning are the backbone of successful mission conduction.

In an analogous manner an Air-to-Ground mission plan can also be seen as a synchronizer. Single aircraft missions with sets of tasks shall encountering a common time frame, and synchronization with other troops and weapon platforms, avoiding no-fly zones and finally cope with general rules of engagements. At a first sight the complexity of all these factors can easily become overwhelming. How can we carry out a mission space parametrization? How can aircraft interact tactically with each other and with other instances, and how can we formulate a mathematical model and a solution method to cover and solve this scope in a reasonable time frame?

The following paper will systematically work through these questions and eventually formulate a framework for an Air-to-ground mission decision support tool, including target sequencing, allocation and routing.

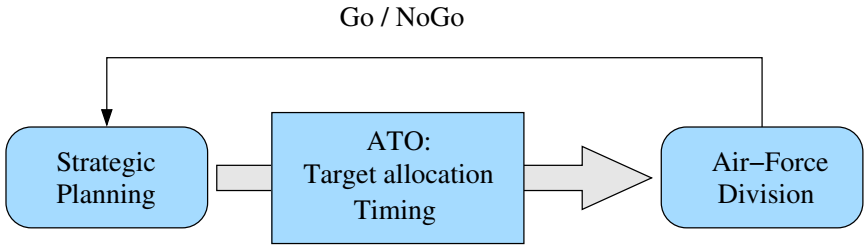


Figure 1: Planning hierarchy where an Air Traffic Order (ATO) is pushed down to the operative level.

Initially as described in Figure 1 the operational planning phase begins upon the receipt of a mission order that includes a detailed description of the target scene, required target effect objectives (e.g. destroyed or neutralized) and exact timing requirements. Tactical information is also described as Forward Line of Own Troops (FLOT), Target scene entrance- and exit points and known surface-to-air missile sites (SAM-sites) as well as protected objects not to be touched by the attacks, like hospitals and schools.

All this information is collectively given in a so called ATO — an Air Task Order. The planning task is now to effectively sequence, allocate and route aircraft according to the ATO with a Go/NoGo feedback and produce plans with high fulfillment of objectives. If an ATO is badly stated the planning activity shall reveal that and return a 'NoGo'. A replanning becomes necessary at the strategic level and eventually a new, more well stated ATO is constructed and delivered. It is a strong must that operational planning not becomes a time lag. Functions and IT systems for fast operational planning is therefore of utmost importance to effectively link the operational and tactical level.

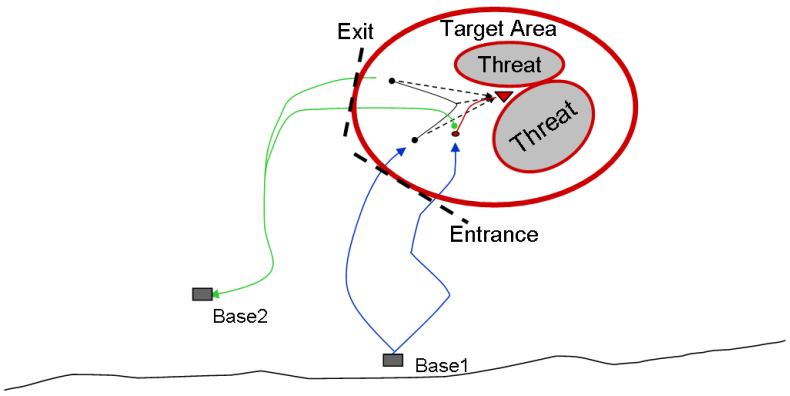


Figure 2: A description of how an ATO can be the main frame of a mission interfaced with the pre mission (en-route) and the post mission (re routing) phases.

Figure 2 shows the target scene, the proposed routing from a deployment Base 1 and the final mission destination at Base 2, the ATO holds data of the target scene, and timing constraints. In order to make an overall plan we will use Effect Oriented Planning (EOP) as a planning doctrine, which means that the effect of a mission is the primary and first planning step. Our objective is to maximize the effect within the target area, hence routing towards and from the target scene is done in an secondary step and will not be analyzed further in this study. However it is straightforward to interface all mission phases based on a successful EOP of the target scene.

1.1 Target Scene

The geographic area of interest, where targets, defenders and protected objects are situated, is referred to as *the target scene*, which is also defined by a line of entrance and a line of exit for the aircraft. A target can be categorized as a specific type such as a house, a bunker, infrastructure or other ground based military objects. The aircraft fleet is deployed from a base positioned on ground or from hangar ships, usually situated far away from the target scene. They enter the scene at the entry line, and when the

mission has been carried out, they leave the scene at the exit line and turn back to the base (or some other base). An example of a small target scene, including three targets and multiple SAMs and protected objects, is found in Figure 3.

The mission time is defined as the time of the first aircraft passing the line of entry until the last aircraft passes the exit line. The diameter of a target scene is usually of the order of 100 km, the distances between targets are of the order of a few kilometers, and the time span of a mission is of the order of one hour. A large attack would involve 6–8 targets and 4–6 aircraft, and would require several hours, at least, of manual planning just to find a feasible attack plan.

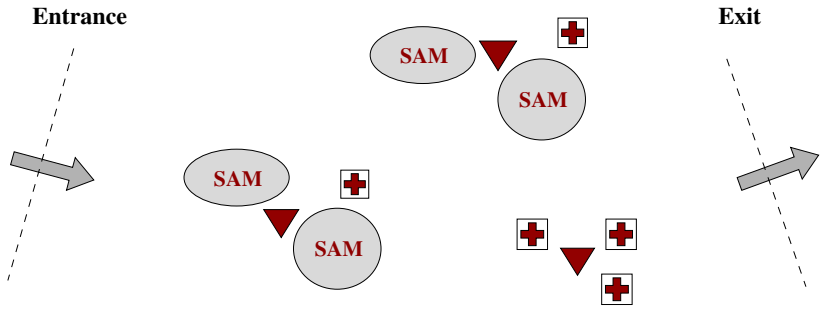


Figure 3: An example of a target scene, including three targets (red triangles) and nearby SAMs and hospitals. The entry line is to the left and the exit line to the right.

Figure 4 shows a target protected by several SAM sites and in a close distance of a protected object. A protected object is an object which must be left untouched during an attack, so risk scenario calculations must be derived due to possible debris or collateral effects in order to "guarantee" safety. Note that terrain, ground and weather (except wind) conditions are left out of this description due to simplicity. Introducing them as effect contributing parameters are of course possible but they will not change the basic principles of how the planning problem is set up and solved.

To handle outer and an inner firing ranges for a specific weapon type, we have parameterized an engagement zone. An outer radius defines the maximum firing range at a specific speed and altitude. An inner radius is related to a too short lock on time for the target seeker. In our description and for simplicity, when firing we assume a standard aircraft cruising speed which makes the engagement zone physically fixed as the geometry of a hollow cylinder.

In order to parameterize the target scene, attack positions are arranged in sectors in accordance with an attack direction. Moreover since a particular weapon system needs guidance to designate a target, specific illumination points are added and tied to each sector. An illuminator is an aircraft with an illumination laser pod system consistently pointing at the target during missile en route and impact.

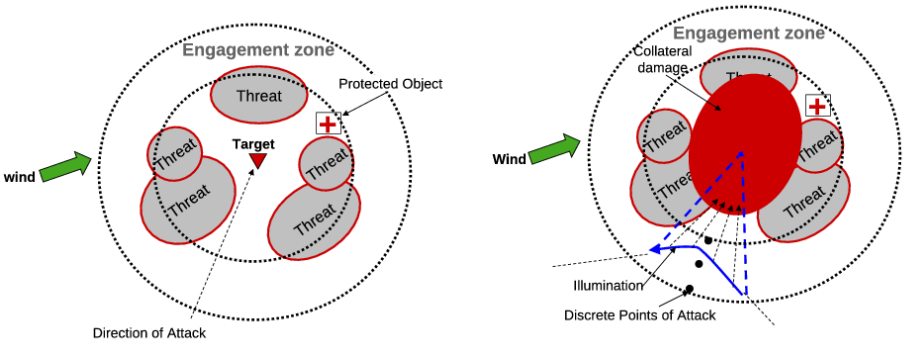


Figure 4: Left: A detailed picture of the target area including primary targets, threats, protected objects and the boundary of the engagement zone. Right: Discrete attack positions and a compatible illumination action. Collateral footprint - does it interfere with protected objects?

As mentioned previously collateral damage must be considered. In Figure 4, a collateral footprint is overlayed. In this case the protected object have some clearance and the proposed attack position can be used. In any case where an attack position causes collateral impact on protected objects these points will be discarded from the possible solution.

In our parameterized engagement zone, our objective is to find the best attack positions with the optimal probability of destroying a target. Basically, effect is measured in each attack position by taking geometric and dynamic considerations.

1.2 Target Effect

Target effect is based on distance, aircraft speed, angle at impact and the degrading fact that a missile path can pass a hostile SAM site zone with an obvious probability of being shot down before impact, see Figure 5. A collateral footprint is defined as an elliptic shape due to fixed angles of deviation which corresponds to an erroneous missile performance.

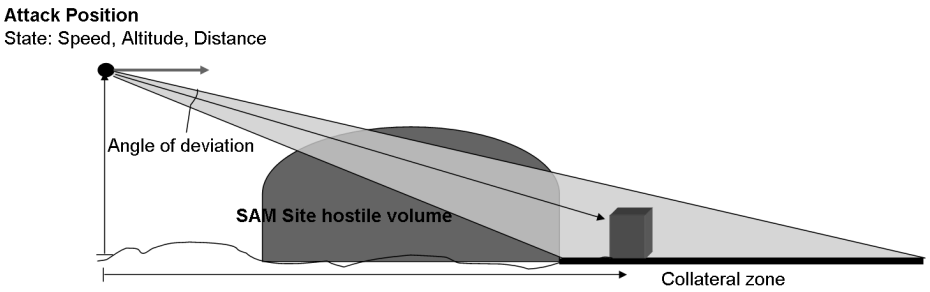


Figure 5: Effect on target as a function of distance, altitude and speed.

An illumination point holds no measure but is the geometric specification of an illumination task as well as it connects the mission path between different tasks. Our mission objective is to destroy a number of targets in a given time frame and with a low risk exposure. Therefore we must strive to find a combination of safe routing and the good choice of high effect attack positions. Thus this is a routing problem with multiple resources and can be addressed as a special case of a Vehicle Routing Problem.

The above discussion provides an understanding of the vital aspects and building blocks of a mission planning scenario. To make the big picture complete, sequencing should be mentioned briefly as the requirements due to some cooperative effect between the targets or due to weather conditions of attacking targets in a specific order.

In this paper it will be seen as preprocessing but the scope can be extended considerably as indicated in Section 1.3.

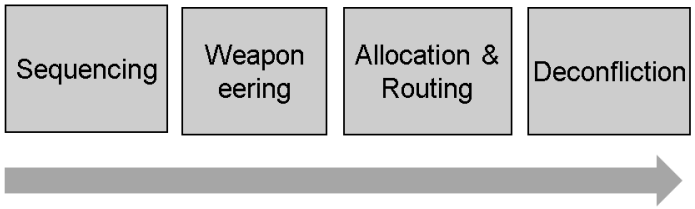


Figure 6: Overall description of functional blocks in the Air-to-Ground Aircraft mission planning problem.

As a summary Figure 6 shows the functional blocks in Air to Ground Aircraft mission planning: Sequencing, Weaponeering, Routing and Deconfliction. Weaponeering is the set up of the target scene with effect measures and collateral effects. Routing is the creation of a network and solving the related routing problem and finally deconfliction holds functionality to make the resulting mission plan conflict-free in time and space.

Since our formulation certainly shows a modular property with distinct and clear interfaces, different parts of the problem can be modeled with different ambitions and granularity. For instance we can thoroughly investigate the sequencing problem putting less effort on the actual mission plan in detail. On the other hand we can disregard sequencing as a forcing constraint and put focus on weaponeering and routing. The last block, Deconfliction covers the task of resolving conflicts in time and space for all resources during the mission plan.

1.3 Sequencing Aspects

Precedence, or the fact that targets has to be orderly processed comes from tactical aspects. In our case we will cover precedence in the whole range from

no precedence up till a predefined target sequence. In this precedence span, what are the tactical aspects to consider? First, in order to use illumination guidance, a target must be fully visible. If debris and dust are stirred up by preceding attacks and transported by wind, the whole attack may fail. As a consequence, wind direction and wind speed is important to consider as well as the expected 'stir-up' effect from an attack. Wind is indicated in Figure 4.

Further there might be connections between targets in a more cooperative manner. In Figure 7-a) a number of SAM sites have been identified as hostile targets, but they cover each other in a way that one sequence might be better than another. If an attack shall be performed with missile paths crossing connected SAM site zones we will be less likely to succeed. So, for the instance in Figure 7-a), a successful strategy is to first remove target I and then target II which removes all sheltering possibilities in any preceding target sequence. This understanding would produce a mission plan with a higher probability of success.

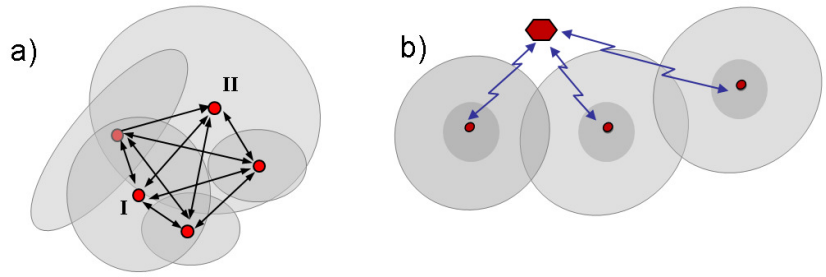


Figure 7: Cooperative sequence dependencies

Another case with sequence dependent costs can be seen in figure Figure 7-b) where a powerful central firing control radar is the backbone for the three SAM sites and their responsiveness and efficiency. If the central radar station is removed first, the three SAM sites will only rely on their own, representing their inner circles, and becomes less agile. Thus, also in this case the analyze of a good sequence is the key to success. There is an obvious possibility of having a target scene including all three of the above types of precedence constraints. In such a mixed environment a thorough sequence analysis is preferred. However in this paper we will make sure that we can process and calculate mission plans based on any kind of sequence constraints, not how these sequences have been identified. Analytical modeling of sequence dependency is referred to as future work. One may mention the most intuitive and simple rule for sequencing in the case of considering wind direction. If we assumes a strong impact from wind drift we can state a main direction of the sequence in the opposite direction, then we will be certain to avoid effect of dust and debris.

1.4 Tactics

Tactics is about how to manage own forces considering both mission objective and outer circumstances. In this paper we have implemented tactics in the model to a certain level and with focus to let the planning personal decide how to operate the mission. Our implementation of models and functions into a mission planning toolkit, promotes tactical decisions in the following cases:

- Sequence: As described in Section 1.3 sequencing and target ordering is a tactical decision. We can handle any kind of sequencing by invoking precedence constraints, any order can be chosen.
- Roles: How each aircraft shall act during a mission is a tactical decision. An aircraft can be dedicated to a certain role such as illumination or attack or both. This can also be decided by the planning personal in the set up of the mission planning tool.
- Tuning the objective: The objective function contains a mix of two mission properties, mission time and hit and destroy probability. How these two mission quantities are emphasized is easily specified by weights and chosen based on tactical aspects.

As can be seen, our mission planning tool handles a range of tactical aspects. However, there are tactical issues that haven't been addressed, but we believe that they can be implemented in a similar way as the previous mentioned.

Additional tactics can be tossing missiles or lobbing missiles, opens tactical flexibility when no air supremacy is the general condition. In that case an aircraft can hide by terrain following or low altitude flight followed by an attack with a lobbed missile path instead of attacking at a higher altitude with an increased risk exposure. Tossing as an attack alternative is implemented by extended the attack positions in the network, one for each attack type. Other tactical aspects are referred to as future work.

1.5 Outline

The paper is structured in the following way. In Section 2, we present a problem overview and put this work into context. The preprocessing and post processing steps are described on a more general level, while the optimization parts are presented in more detail.

In Section 3 we present the actual problem at hand in more detail, and discuss some modeling issues and problem limitations. In Section 4 we introduce notations and present a mathematical model and point out the relationship to the Generalized Vehicle Routing Problem (GVRP). An alternative model formulation is also given, which require less binary variables.

In order to validate the mathematical models, Section 5 provide solutions to a small test case scenario. A comparison in runtimes between the two formulations are also presented.

Section 6 give a possible transformation of a Generalized multiple Traveling Salesmen Problem (GmTSP) into a standard Traveling Salesman Problem (TSP). There is also information on the Lin-Kernighan algorithm, a powerful and efficient heuristic for TSP problems that can be used to find near-optimal solutions. These transformations are used in the upcoming sections, where efficient heuristics for the aircraft mission planning problem are discussed.

Sections 7 and 8 present two heuristic approaches for the problem at hand. The first heuristic is a two-step constructive heuristic, based on the transformations of GmTSP problems into standard TSP problems discussed in the previous section. The second heuristic is inspired by column generation, but is also close to a total enumeration method. Due to the limited number of targets in a target scene, it is possible to generate optimal solutions for all subsets of targets, and then finding near-optimal solutions by fitting these "optimal columns" together. The master problem becomes a Set Partitioning Problem (SPP) with a limited number of columns.

Section 9 present a benchmark for the proposed heuristics, together with results from the mathematical model, followed by conclusions and future work in Section 10.

2 Problem Overview

In order to put our work into a context, Figure 8 provides a flow chart with dependencies and subproblems for the three main parts: Preprocessing, Optimization and Post processing.

All parts are equally important for this kind of work, military mission planning, if it is to be used in practice. If we do not model reality good enough, the solutions are not meaningful. In the same way, even though we are able to find optimized solutions, they are not very interesting if impossible to use in practice. With that said, the optimization process is the most interesting and most challenging part, and thus our main focus in this paper.

Each box in the flow chart is by itself a challenging problem, and we have not dealt with all of them. Dashed lines indicate parts that are subject to future work, while full lines indicate parts that are described in more detail.

We only consider the *Feasible Attack Space*-box in the preprocessing step, as it provides input data for the model. The other boxes are complex enough for a full paper each, hence set aside at the moment. For the work in the optimization step, it is important to point out that we assume airspace supremacy at the moment, focusing on the routing aspects. It should be

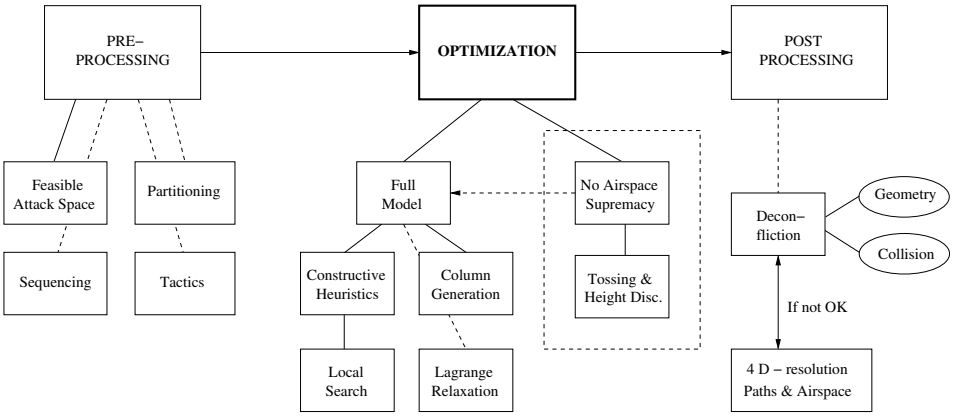


Figure 8: A flow chart of the problem structure. The main three boxes contain several sub-boxes, each an interesting problem, and dashed lines indicate parts not yet implemented.

straightforward to extend the mathematical model for the non-supremacy case, at the cost of more variables and constraints. The Post processing box is not available at the moment.

2.1 Preprocessing

The preprocessing step is necessary in order to generate the data needed for a mathematical model, and is broken down into smaller parts in accordance with Figure 8. Here follows a short description of each box.

Feasible Attack Space

This is a vital part of the preprocessing, where the so called feasible attack positions are located and evaluated. For an attack position to be feasible, there cannot be any risk of collateral damage for the protected objects in the vicinity of the target. Ingoing details are found in Section 3.3, where the attack space is discretized and a network representation of the problem is created.

Partitioning

Geometry is a big part of the problem nature, and for any problem instance, targets are clustered based on their geographical locations. The only criteria for these clusters should be that all distances between clusters widely exceeds that of inter-cluster targets, hence making it uninteresting to solve the mission planning problem for all targets at once. For the example shown in Figure 9, the ten targets have been partitioned into three clusters, each posing a smaller problem that can be solved independently of the others.

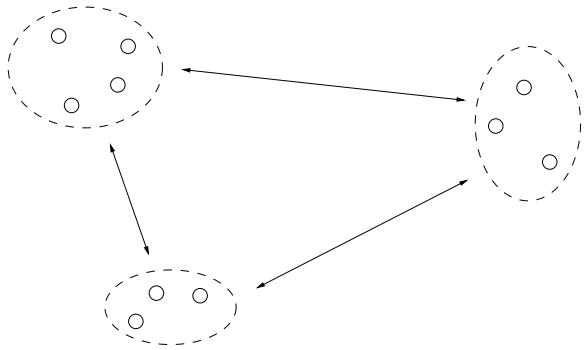


Figure 9: An a priori partitioning of targets into smaller problems to be solved independently.

The aircraft sometimes need to return to their base for refueling and rearmouring in between the clusters, but otherwise one can solve a small traveling salesman problem problem to decide in which order the clusters should be attacked. Alternatively, the resources are split up in order to attack all clusters simultaneously. It is also possible that an attack order is decided manually for other reasons.

Sequencing and Tactics

Sequencing is an important preprocessing step. Due to strong wind conditions, a natural sequence might be necessary in order to avoid stirring up dirt which will drive away and cover other targets, making it impossible to illuminate and attack them. Other factors might be the result of an analysis of mutual dependencies, allowing for much greater effect towards targets or increased survival rate. See Section 1.3 for a more detailed discussion about sequencing.

Tactics involve issues that are difficult to model and quantify, like experience and human opinions, and hence must be settled before the mathematical framework can be used. A thorough discussion is found in Section 1.4.

2.2 Optimization

The optimization box is the main focus of the paper and ingoing details are found throughout Sections 3–8. At this point, we assume airspace supremacy, which means that no enemy aircraft is present. This allows for only considering flight paths on relatively high altitude, high enough to avoid all threats on the ground. It means that we only need to consider one altitude layer, i.e. making the problem 2-dimensional. Future work is to adjust the proposed methods to handle the situation where airspace is not secured.

Another thing that becomes relevant in the situation of no airspace supremacy is the tactical aspect for the attacker. If airspace is not secured, a resource would want to take advantage of the surroundings by approaching the target from a direction where for example high hills and tall buildings protect it from being detected. This could mean flying at low altitude, and if so the missile should be "tossed in" towards the target rather than being launched straight towards it.

A full mathematical model is presented in Section 4 together with a discussion about possible extensions and ways to strengthen the model. The model is validated in Section 5 which includes illustrative examples of typical solutions. Direct solution with a general purpose MIP solver is efficient only for problems of moderate size though, and we propose and implement two heuristic approaches, Constructive Heuristics described in Section 7 and a Column Generation inspired approach found in Section 8.

2.3 Post processing

The post processing consists of a deconfliction step, which is necessary in order to see if the proposed flight paths are realizable, i.e. safe and sound from a pilots view. Any collision courses should be resolved, either by separating the conflicting paths in space (altitude) or delay one of the paths in time.

The deconfliction step is very important in order to actually use the attack plan proposed by the optimization procedures. Details are discussed more thorough in Section 9.8, but this part is mainly on a more theoretical level at the moment and subject to future work.

3 Modeling Issues

In order to model the complex reality of the problem, we need to do some assumptions and simplifications. In this section we describe how different aspects of the problem formulation is handled.

3.1 Attack and illumination

An attack requires two aircraft to team up, where one of them illuminates the target with a laser beam and the other launches the weapon (bomb or missile). We assume that the flight direction of the aircraft is directed towards the target at the time of the launch.

The illumination is required in order to guide the weapon towards its target, providing high accuracy of the impact, and it needs to be continuously visible for the weapon. This means that the aircraft need to rendezvous not only in time, but also in such a way that the illumination is visible for the attacker

at the launch of the weapon. Hence, the illumination begins shortly before the launch of the weapon and has to continue until impact. A typical flight path for an attacker, and a flight path for an aircraft illuminating a target, can be found in Figure 10.

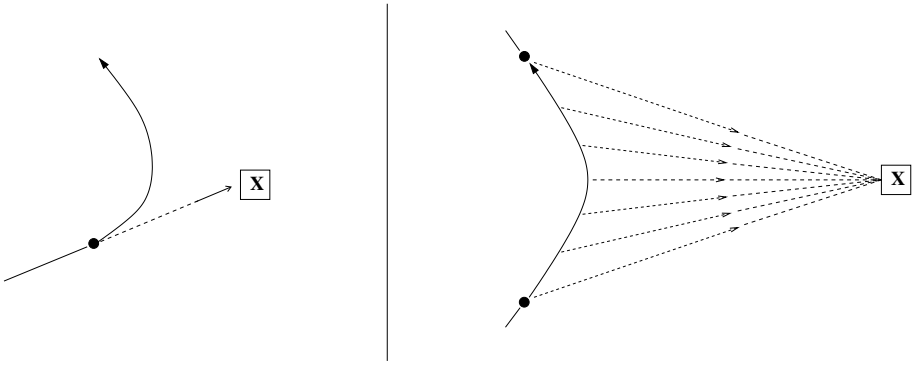


Figure 10: Left: Attack path. The aircraft flies towards the target, launches the weapon, and makes an evasive maneuver. Right: Illumination path. From its starting position, the aircraft flies in a parabolic path and illuminates the target continuously.

When a target is attacked, the air around it will be filled with dust and debris, and due to the prevailing wind conditions this might reduce the visibility of nearby targets, hence it is realistic to assume that some precedence constraints are given, specifying which targets that are not allowed to be attacked before other targets.

3.2 Restrictions and limitations

The expected effect of an attack depends, of course, of the kind of weapon being used, which is decided in advance, but also of the direction of the impact and its kinetic energy. The latter factors depend on the velocity and altitude of the aircraft at the time of the launch. Further, if the weapon passes through defended airspace, its expected effect is reduced.

No matter how accurate the attack can be performed, the neighbouring area of the target is always subject to a certain risk of collateral damage, because the weapon can miss its target. This can be due to for example loss of visibility of the illumination, malfunction of the weapon or defense measures.

We refer to the area of unacceptably high risk for collateral damage as the *footprint* of the attack, visualized in Figure 11, and it depends on the altitude and velocity of the attacker. The footprint is in our description of the problem simply given by a straight line from the attack position towards the target, and an angle of maximal deviation from this line. This construction gives an ellipsoid-shaped footprint on the ground.

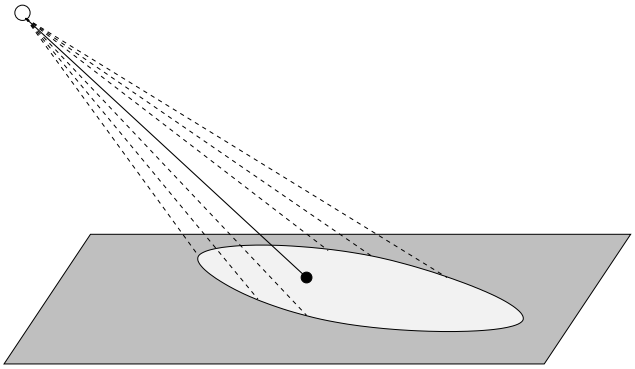


Figure 11: Footprint of an attack position. The black line represents the correct path of the missile, and the dotted lines its maximum deviations.

We define an attack position to be feasible if no protected object is inside its footprint. For a given target, and an aircraft with specified characteristics such as velocity, altitude and armament, one can derive a region of feasible attack positions, referred to as the feasible attack space for that target. Each attack position in this space is associated with a number of feasible illumination positions, where one illumination position can be compatible with multiple attack positions.

Each aircraft has a limited armament capacity, meaning that it cannot carry more than a certain amount of weapons, which limits the number of attacks it can perform. In addition to the armament, an aircraft can also carry an illumination laser pod. Without the illumination pod, an aircraft can only perform attacks. An aircraft might also be equipped with the illumination laser pod only, hence only capable of performing illumination.

Note that once the planning of the mission has been made, it is also known how each and one of the aircraft shall be equipped in order to be able to fulfill its tasks during the mission.

3.3 Feasible Attack Space

For a specific type of aircraft and a target requiring a specific type of weapon, one can derive the feasible attack space against the target, here represented by an inner and an outer radius of attack plus an upper and a lower altitude, that is, an attack space that can be visualized as a hollow cylinder. A two-dimensional projection of this cylinder onto the ground is found in Figure 12.

This hollow cylinder is divided into altitude layers and into a number of sectors, in which we discretize feasible attack positions. For each and one of these we create compatible illumination positions. Since only feasible attack positions are included, and these depend on the presence of protected objects and air defense, the number of such positions in each sector might vary.

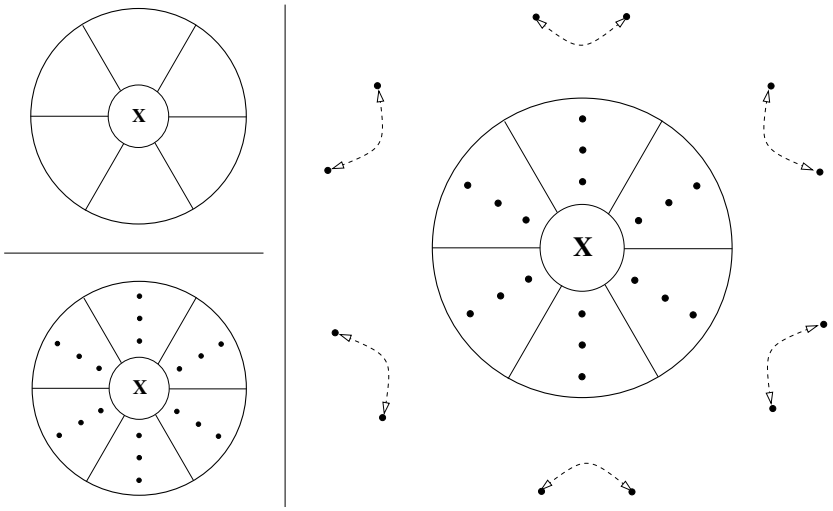


Figure 12: Upper left: The feasible attack space as a hollow cylinder, divided into sectors. Lower left: A coarse discretization of feasible attack positions in each sector. Right: Compatible illumination alternatives for each sector. Each illumination alternative is compatible with all attack positions in the same sector.

We have chosen to introduce three discrete attack positions in each sector and on each altitude. For these three positions we introduce two common illumination alternatives. For both illumination alternatives, the illumination of the aircraft goes on during the flight from a starting position to an ending position. The two illumination alternatives are obtained by interchanging the roles of these two positions and reversing the flight direction. The two positions are chosen so that the illuminating aircraft is flying essentially clockwise or counter-clockwise in relation to the target. See Figure 12 for an illustration of this.

3.4 Network representation

By performing a discretization of the feasible attack space around each target, representing attack and illumination positions by nodes, and aircraft movements by arcs, the mission planning problem can partly be represented by a network. A dummy origin and a dummy destination are introduced to represent the crossing of the entry and exit lines of the target scene, respectively.

Each target shall be attacked and illuminated exactly once, and an aircraft can not both attack and illuminate a target. Hence, the network only contains arcs between nodes corresponding to different targets, or from the dummy origin or to the dummy destination. The exception are arcs connecting the starting and ending nodes of each illumination alternative, which are also included in the network although they correspond to the same target.

Moreover, no arcs should violate any given precedence constraints between targets. Hence there might be arcs from attack nodes for target 1 towards attack and illumination nodes for target 2, but not the other way around, depending on the given precedence relations.

On a more abstract level, nodes in the network can be clustered and represented as in Figure 13, where each target is associated with two clusters, one containing all attack nodes (A) and the other containing illumination nodes (I). The exact structure of these clusters is found in Figure 14.

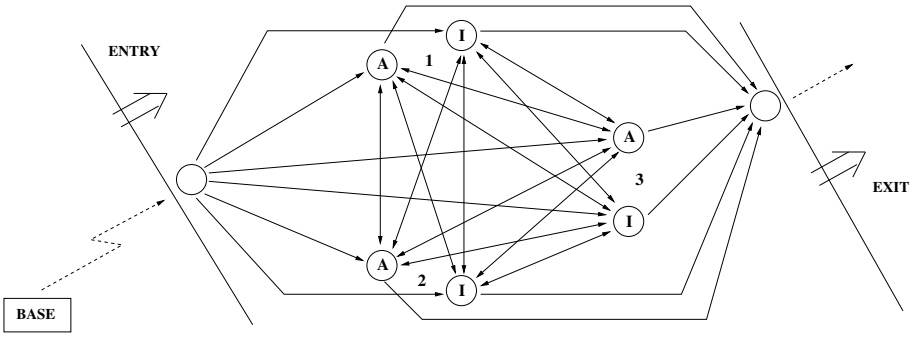


Figure 13: Given a base and three targets, the aircraft fleet should visit each cluster (A=attack and I=illumination) exactly once. One aircraft is not allowed to both attack and illuminate the same target though.

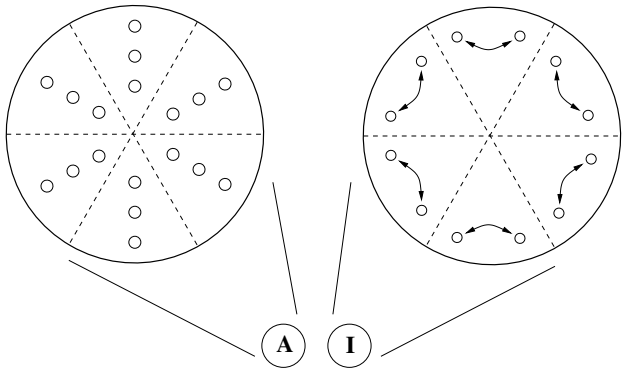


Figure 14: Structure of the clusters. An attack cluster consists of many attack positions, and exactly one should be visited. The same goes for the illumination clusters. Also, the visited attack and illumination positions need to belong to the same sector.

3.5 Arc costs

For calculating the arc costs in the network representation of the problem, we must find flight distances between all candidate positions. A feasible path between two positions (nodes) is a path where the restrictions of the aircraft

dynamics is taken into account, such as turning radius and other physical limitations. The path also needs to be safe, meaning that the aircraft cannot pass through defended airspace.

In the literature, the problem of finding an optimal flight path from a given starting point to a given destination, while avoiding obstacles, such as defended airspace, is referred to as the Aircraft Routing Problem. This is in itself a difficult optimization problem, not addressed in this paper, and we refer to [13] and [2] as examples of algorithms that can be used to solve this problem. A closely related routing problem is described in [5] and [12], which gives rise to a shortest-path problem with side constraints.

In our numerical experiments we used a flight path generator provided by our industrial partner. It takes aircraft dynamics into account and is based on a discretization of the airspace and a calculation of a shortest path.

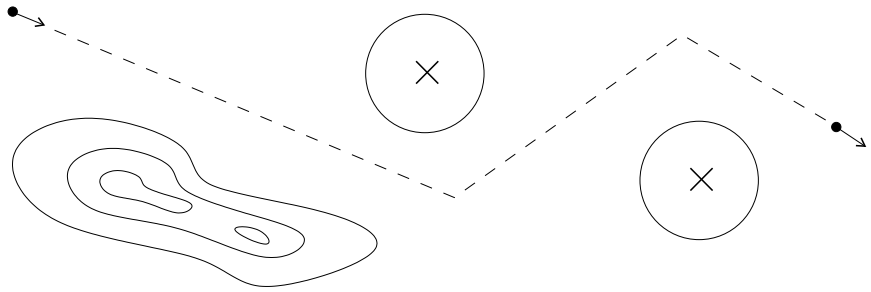


Figure 15: An example of aircraft routing. The problem is to find a flight path from a given starting point to a given destination point, avoiding obstacles and defended airspace.

The result of each such routing problem is a feasible path, with a certain length, which can be converted into a time required to traverse it. Note that the arc lengths and traveling times obtained are not symmetric. In our network representation, the nodes are associated with *both* a location and a flight direction at the location. Because of these flight directions and the flight dynamics, the path lengths and path times will in general be asymmetric.

In addition to the time attribute, each arc leaving an attack position also has an attribute that states the expected effect against the target associated with the attack position.

4 Mathematical Description

4.1 Notation

Given is an aircraft fleet $\mathcal{R}$, and a set of targets $\mathcal{M}$ to be attacked. Each target $m \in \mathcal{M}$ is associated with a feasible space of attack positions, discretized into attack positions N_m^A, and their compatible illumination positions, N_m^I. Furthermore, each feasible attack space is divided into G sectors, and we let $\mathcal{G}$ denote the set of all sectors for all targets while $\mathcal{G}_m$ is the set of sectors that belong to target $m \in \mathcal{M}$.

Let A_g and I_g denote the set of arcs (i, j) such that position j is an attack position and illumination position respectively in sector g, for all sectors $g \in \mathcal{G}$. Further, let AI_g denote the set of arcs (i, j) such that position j is either an attack position or an illumination position for all sectors $g \in \mathcal{G}$. Furthermore, let N denote all positions in the graph, including dummy origin and dummy destination, while N^* denotes the set of all positions except the origin and the destination. Also, let A denote all arcs in the network. Each aircraft $r \in \mathcal{R}$ is limited to carry at most Γ weapons, and let q_m denote the number of weapons needed towards target $m \in \mathcal{M}$.

Let $\mathcal{S}$ denote the set of ordered pairs (m, n) of targets such that target m cannot be attacked before target n. If no precedence relations are given a priori, the set $\mathcal{S}$ is empty. Let c_{ij}^r denote the cost of arc (i, j) for aircraft r. For arcs leaving attack positions, that is $i \in N_m^A,\ m \in \mathcal{M}$, the value of c_{ij}^r is the expected effect of the attack. Otherwise, c_{ij}^r takes the value zero. Further, let T_{ij}^r denote the time needed for aircraft $r \in \mathcal{R}$ to traverse arc $(i, j) \in A$. We also introduce T_{max}, either as a pessimistic estimate of the total mission time or as a given upper time limit for the duration of the mission.

We introduce two types of variables, the binary routing variables x_{ij}^r and the continuous time variables t_i^r, t_m^A, t_m^I, and t_F. The routing variable x_{ij}^r equals one if aircraft $r \in \mathcal{R}$ traverses arc $(i, j) \in A$, otherwise it is zero. Variable t_i^r is the time at which aircraft $r \in \mathcal{R}$ visits node $i \in N$ and it is equal to zero if the aircraft does not visit the node. Variables t_m^A and t_m^I are the times of the attack and illumination, respectively, of each target $m \in \mathcal{M}$, and t_F is the time of the last aircraft to exit the target scene.

4.2 The Model

The mathematical model for the Military Aircraft Mission Planning Problem (MAMPP) is given below.

$$\max \sum_{r\in\mathcal{R}}\sum_{(i,j)\in A} c_{ij}^r x_{ij}^r \; - \; \mu t_F \qquad [MAMPP]$$

$$s.t. \sum_{(o,j)\in A} x_{oj}^r = 1, \quad r\in\mathcal{R} \qquad (1a)$$

$$\sum_{(i,d)\in A} x_{id}^r = 1, \quad r\in\mathcal{R} \qquad (1b)$$

$$\sum_{(i,k)\in A} x_{ik}^r - \sum_{(k,j)\in A} x_{kj}^r = 0, \quad r\in\mathcal{R},\ k\in N^* \qquad (2)$$

$$\sum_{r\in\mathcal{R}}\sum_{g\in\mathcal{G}_m}\sum_{(i,j)\in A_g} x_{ij}^r = 1, \quad m\in\mathcal{M} \qquad (3)$$

$$\sum_{r\in\mathcal{R}}\sum_{g\in\mathcal{G}_m}\sum_{(i,j)\in I_g} x_{ij}^r = 1, \quad m\in\mathcal{M} \qquad (4)$$

$$\sum_{r\in\mathcal{R}}\sum_{(i,j)\in A_g} x_{ij}^r - \sum_{r\in\mathcal{R}}\sum_{(i,j)\in I_g} x_{ij}^r = 0, \quad g\in\mathcal{G} \qquad (5)$$

$$\sum_{g\in G_m}\sum_{(i,j)\in AI_g} x_{ij}^r \le 1, \quad r\in\mathcal{R},\ m\in\mathcal{M} \qquad (6)$$

$$\sum_{m\in\mathcal{M}}\sum_{g\in\mathcal{G}_m}\sum_{(i,j)\in A_g} q_m x_{ij}^r \le \Gamma, \quad r\in\mathcal{R} \qquad (7)$$

$$t_i^r + T_{ij}^r x_{ij}^r - T_{max}(1-x_{ij}^r) \le t_j^r, \quad r\in\mathcal{R},\ (i,j)\in A \qquad (8)$$

$$t_i^r \; - \; T_{max}\sum_{(i,j)\in A} x_{ij}^r \le 0, \quad r\in\mathcal{R},\ i\in N \qquad (9)$$

$$t_o^r = 0, \quad r\in\mathcal{R} \qquad (10)$$

$$\sum_{r\in\mathcal{R}}\sum_{i\in N_m^A} t_i^r = t_m^A, \quad m\in\mathcal{M} \qquad (11)$$

$$\sum_{r\in\mathcal{R}}\sum_{i\in N_m^I} t_i^r = t_m^I, \quad m\in\mathcal{M} \qquad (12)$$

$$t_m^A = t_m^I, \quad m\in\mathcal{M} \qquad (13)$$

$$t_m^A \ge t_n^A, \quad (m,n)\in\mathcal{S} \qquad (14)$$

$$t_F \ge t_d^r, \quad r\in\mathcal{R} \qquad (15)$$

$$x_{ij}^r \in \{0,1\}, \quad r\in\mathcal{R},\ (i,j)\in A \qquad (16)$$

$$t_i^r \ge 0, \quad r\in\mathcal{R},\ i\in N \qquad (17)$$

$$t_m^A,\ t_m^I \ge 0, \quad m\in\mathcal{M} \qquad (18)$$

$$t_F \ge 0, \qquad (19)$$

The objective is to maximize the expected effect against all targets, weighted against the total mission time, that is the time of the last aircraft to pass the exit line, by parameter $\mu \geq 0$. Constraints (1) and (2) ensure that each aircraft enters and leaves the target scene via the dummy nodes, and (3) ensures that each target $m \in \mathcal{M}$ is attacked by exactly one aircraft, and (4) does the same for illumination.

Constraint (5) ensures that the attack and the illumination against each target are compatible, that is, that the nodes belong to the same sector. In a sector where no attack is performed, no illumination can be performed either, and vice versa.

Constraint (6) states that each aircraft can visit each target at most once. This constraint is actually redundant since the time propagation constraint (8) together with the synchronization constraint (13) make it impossible for one resource to both attack and illuminate the same target, but it results in a model with a more tight linear programming relaxation. Constraint (7) is the armament capacity constraint and limits each aircraft to utilize at most Γ missiles.

Constraint (8) propagates time for each aircraft, making sure that if aircraft r traverses arc (i, j), node j is visited no earlier than the time of the visit to node i plus the time needed to traverse the arc. Note that constraint (8) also eliminates subtours. Constraint (9) enforces that $t_i^r = 0$ holds if node i is not visited by aircraft r, and (10) states that all aircraft start from the origin at time zero.

Constraints (11) and (12) assign the correct times of attack and illumination, respectively, for each target m, and constraint (13) states that these times need to coincide, that is synchronization in time. Constraint (14) makes sure that targets are attacked in the prescribed precedence order. It is possible to eliminate variables t_m^A and t_m^I from the model, but they are kept for the sake of readability.

Note that although the network does not contain arcs that violate the precedence relationships, constraint (14) is still needed to fully prevent violation of these relationships. This is so because the attack sequences of two aircraft might together violate precedence, even though each of them does not. Finally, constraint (15) finds the maximum time of return to the destination node among all aircraft, to be used in the objective.

4.3 Characteristics of the model

The model presented above belongs to the class of Vehicle Routing Problem (VRP). Attack and illumination points are nodes, and paths between such positions are arcs. The aircraft fleet correspond to resources with capacity constraints on their weapon load, and targets correspond to customers. The model has the following non-standard characteristics.

i) It is generalized in the respect that exactly one node in each cluster shall be visited.

ii) Since the attack and illumination positions for a target need to be compatible, the visits to attack and illumination clusters are coupled by side constraints.

iii) The visits to the compatible attack and illumination nodes for a target are required to be exactly synchronized in time.

iv) The order in time of the visits to the pairs of attack and illumination clusters of all targets are constrained by precedence relations.

In the Generalized Traveling Salesman Problem, the nodes are partitioned into clusters and the salesman shall visit exactly one node in each cluster, at minimum cost. This problem has been studied to some extent, see for example [8] and [11]. The corresponding generalization of the Vehicle Routing Problem (GVRP) has been studied much less. To our knowledge, the first to discuss this problem are Ghiani and Improta [6], who give a transformation to the Capacitated Arc Routing Problem (CARP). Baldacci *et al.* [1] discuss some applications of the GVRP. Formulations and branch-and-cut algorithms for the GVRP are given in the recent paper [4].

To the best of our knowledge, the military aircraft mission planning problem presented here have not earlier been modeled as a generalized vehicle routing problem with compatability of visits, synchronization in time, and precedence relations.

4.4 Extending the model

Due to the presence of T_{max} in constraints (8) and (9), the linear problem relaxation is weak. For the specific instances of the problem we want to solve, it is possible to strengthen the model. Since scenarios including eight targets are considered to be large instances, one can introduce an extra binary variable, u^r_{mn}, that equals one if aircraft r travels directly from target m to target n, and zero otherwise. These variables are defined on the set of ordered pairs $A_{\mathcal{M}} = (\mathcal{M} \times \mathcal{M}) \setminus \mathcal{S}$. The x^r_{ij} and u^r_{mn} variables are coupled by

$$\sum_{i \in N_m, j \in N_n} x^r_{ij} = u^r_{mn}, \qquad r \in \mathcal{R},\ (m,n) \in A_{\mathcal{M}}$$

where N_m denotes all nodes connected to target m.

Even for a scenario with ten targets, there are only $K = 2^{10} - 1 = 1023$ nonempty subsets of targets S_k, used to define subtour eliminating constraints with respect to the new variables u^r_{mn}. By adding them all, the model is strengthened significantly.

$$\sum_{m \in S_k, n \in S_k} u^r_{mn} \leq |S_k| - 1, \qquad r \in \mathcal{R},\ k = 1, \ldots, K$$

Also, as always in VRP problems where resources are identical, symmetry is an issue. It is possible, without loss of generality, to add constraints stating that the first target is attacked by a specific aircraft and illuminated by another specific aircraft. For the special case of only two aircraft, one can also add constraints forcing them to traverse the targets in the same order, i.e. enforcing $u^1_{mn} = u^2_{mn}$.

The aircraft fleet can of course be required to be utilized in different ways. For example, an aircraft r_1 can be specified to operate pairwise with another aircraft r_2 throughout the mission, which is now easily modeled as $u^{r_1}_{mn} = u^{r_2}_{mn}$. An aircraft r can also be given a specified role throughout the mission, that is, performing attacks *or* illuminations only. This is modeled by $x^r_{ij} = 0$ for all $(i,j) \in I_g,\ g \in \mathcal{G}$ and $x^r_{ij} = 0$ for all $(i,j) \in A_g,\ g \in \mathcal{G}$ respectively. Both assumptions, working pairwise and specified roles, can be utilized simultaneously.

4.5 MIP Model v2

A second MIP model for the MAMPP problem has been developed. Instead of routing variable x^r_{ij} between all nodes, it consists of routing variable u^r_{mn}, as defined earlier, and binary variables y^r_i that is equal to one if node i is visited by resource r, and zero otherwise. This reduces the number of binary variables drastically.

We introduce the following notation. For each sector $g \in \mathcal{G}$, denote its attack positions N^A_g, and their compatible illumination positions, N^I_g. We also need to find tuples (i,j,m,n) such that node i and node j belongs to target m and target n, respectively, as well as arc (i,j) exist and aircraft are allowed to go directly from target m to target n.

$$IDX \ := \ \{(i,j,m,n) \mid (m,n) \in A_{\mathcal{M}},\ (i,j) \in A \ :\ i \in N_m,\ j \in N_n\}$$

This model is formulated as a minimization problem, although the objective is of course still to maximize the expected effect against all targets, weighted against the total mission time. A new auxiliary variable α_{ij} is also needed in order to model an equivalent objective function. This variable becomes nonzero exactly when a resource r travels between targets m and n, and one variable y^r_i is active at both targets, that is, equivalent to our old variable x^r_{ij}.

$$\min \quad \sum_{(i,j)\in A_r} \alpha_{ij} \ + \ \mu t_F \qquad\qquad [MAMPPv2]$$

$$s.t. \quad \sum_{(o,n)\in A^r_{\mathcal{M}}} u^r_{on} \ = \ \sum_{(m,d)\in A^r_{\mathcal{M}}} u^r_{md} \ = \ 1 \qquad r \in \mathcal{R} \qquad (1)$$

$$\sum_{(m,k)\in A^r_{\mathcal{M}}} u^r_{mk} \ - \ \sum_{(k,n)\in A^r_{\mathcal{M}}} u^r_{kn} \ = \ 0 \qquad r \in \mathcal{R},\ k \in \mathcal{M} \qquad (2)$$

$$\sum_{\substack{m\in S_k, n\in S_k:\\ (m,n)\in A_{\mathcal{M}}}} u^r_{mn} \leq |S_k|-1 \qquad r\in\mathcal{R},\ k\in\mathcal{K} \tag{3}$$

$$\sum_{r\in\mathcal{R}}\sum_{i\in N^A_m} y^r_i = 1 \qquad m\in\mathcal{M} \tag{4a}$$

$$\sum_{r\in\mathcal{R}}\sum_{i\in N^I_m} y^r_i = 1 \qquad m\in\mathcal{M} \tag{4b}$$

$$\sum_{r\in\mathcal{R}}\sum_{i\in N^A_g} y^r_i - \sum_{r\in\mathcal{R}}\sum_{i\in N^I_g} y^r_i = 0 \qquad g\in\mathcal{G} \tag{5}$$

$$\sum_{i\in N_m} y^r_i \leq 1 \qquad r\in\mathcal{R},\ m\in\mathcal{M} \tag{6}$$

$$\sum_{m\in\mathcal{M}}\sum_{i\in N_m} q_m y^r_i \leq \Gamma, \qquad r\in\mathcal{R} \tag{7}$$

$$\sum_{(m,k)\in A^r_{\mathcal{M}}} u^r_{mk} = \sum_{(k,n)\in A^r_{\mathcal{M}}} u^r_{kn} = \sum_{i\in N_k} y^r_i \qquad r\in\mathcal{R},\ k\in\mathcal{M} \tag{8}$$

$$y^r_k = 1 \qquad r\in\mathcal{R},\ k\in\{o,d\} \tag{9}$$

$$t^r_i + T^r_{ij}\cdot(y^r_i + y^r_j + u^r_{mn} - 2)$$

$$-T_{max}\cdot(3-y^r_i-y^r_j-u^r_{mn}) \leq t^r_j \qquad r\in\mathcal{R},\ \binom{i,j}{m,n}\in IDX \tag{10}$$

$$t^r_i - T_{max}\cdot y^r_i \leq 0 \qquad r\in\mathcal{R},\ i\in N \tag{11}$$

$$t^r_o = 0 \qquad r\in\mathcal{R} \tag{12}$$

$$c^r_{ij}\cdot(y^r_i + y^r_j + u^r_{mn} - 2) \leq \alpha_{ij} \qquad r\in\mathcal{R},\ \binom{i,j}{m,n}\in IDX \tag{13}$$

$$\sum_{r\in\mathcal{R}}\sum_{i\in N^A_m} t^r_i = t^A_m \qquad m\in\mathcal{M} \tag{14}$$

$$\sum_{r\in\mathcal{R}}\sum_{i\in N^I_m} t^r_i = t^I_m \qquad m\in\mathcal{M} \tag{15}$$

$$t^A_m = t^I_m \qquad m\in\mathcal{M} \tag{16}$$

$$t^A_m \geq t^A_n \qquad (m,n)\in\mathcal{S} \tag{17}$$

$$t_F \geq t^r_d \qquad r\in\mathcal{R} \tag{18}$$

$$u^r_{mn} \in \{0,1\} \qquad r\in\mathcal{R},\ (m,n)\in A_{\mathcal{M}} \tag{19}$$

$$y^r_i \in \{0,1\} \qquad r\in\mathcal{R},\ i\in N \tag{20}$$

$$z_g \in \{0,1\} \qquad g\in\mathcal{G} \tag{21}$$

$$t^r_i \geq 0 \qquad r\in\mathcal{R},\ i\in N \tag{22}$$

$$t^A_m,\ t^I_m \geq 0 \qquad m\in\mathcal{M} \tag{23}$$

$$t_F \geq 0, \tag{24}$$

$$\alpha_{ij} \geq 0 \qquad (i,j)\in A \tag{25}$$

Constraints (1) and (2) ensure that each aircraft enters and leaves the target scene via the dummy nodes, and (3) is the subtour eliminating constraints. Constraint (4a) ensures that each target $m \in \mathcal{M}$ is attacked by exactly one aircraft, and (4b) does the same for illumination. Constraint (5) ensures that the attack and the illumination against each target are compatible, that is, that the nodes belong to the same sector.

Constraint (6) states that each aircraft can visit each target at most once, and (7) is the armament capacity constraint and limits each aircraft to utilize at most Γ missiles. Constraint (8) couples variables u^r_{mn} and y^r_i, and (9) states that each aircraft must leave the origin and return to the destination. Constraint (10) propagates time for each aircraft, making sure that if aircraft r visits node i at target m directly followed by a visit at node j at target n, node j is visited no earlier than the time of the visit to node i plus the time needed to travel between the nodes. Note that constraint (10) also eliminates subtours. Constraint (11) enforces that $t^r_i = 0$ holds if node i is not visited by aircraft r, and (12) states that all aircraft start from the origin at time zero.

Constraint (13) defines the value of the auxiliary variable α_{ij}, so that it becomes equal to the cost of traveling between nodes i and j exactly when a resource r travels between targets m and n, that is, variables u^r_{mn}, y^r_i and y^r_j are all active. Constraints (14) and (15) assign the correct times of attack and illumination, respectively, for each target m, and constraint (16) states that these times need to coincide, that is synchronization in time. Constraint (14) makes sure that targets are attacked in the prescribed precedence order. It is possible to eliminate variables t^A_m and t^I_m from the model, but they are kept for the sake of readability.

Constraint (17) prevent violation of the precedence relations, and (18) finds the maximum time of return to the destination node among all aircraft, to be used in the objective. This second model is interesting since it contains much less binary variables than the first model.

5 Empirical Testing

In this section we present some numerical results for a small-size test case scenario, in order to verify the mathematical models presented in the previous section, and to illustrate characteristics of solutions. The scenario described in Figure 3 is solved for two aircraft, using the MAMPP model, based on the network representation shown in Figure 13. All numerical data used in the scenario was provided by our industrial partner.

We use one altitude layer, divided into $G = 6$ sectors, with three attack positions and two illumination positions in each, that is, the clusters shown in Figure 14, which results in a network model including 67 nodes and 2830 arcs. (Some nodes are removed due to protected objects, hence all sectors are not used.) We use $\mu = 0.1$ in the objective, and parameter value $q_m = 1$

for all targets. The model was implemented using AMPL, and the tests were performed on a HP DL160 server with two 6-core Intel Xeon CPUs and 72 GB of RAM memory, running Linux, and the MIP solver used were CPLEX/12.2 for 64 bit environment.

5.1 Default settings

In the baseline case, there are no given precedence constraints, and $\Gamma = 3$, allowing one resource to attack all three targets. After AMPL and CPLEX preprocessing, the problem consists of 4468 rows, 4273 columns and 4155 binary variables, and it was solved in 359 seconds. The result is presented in Figure 16.

The attack sequence becomes 2–1–3, where one aircraft performs all attacks and the other aircraft illuminates the targets, which results in a total mission time of $t_F = 289$ seconds. The expected effect of the attacks against targets 2 and 3 are maximal, among the available attack positions for these targets, while the attack position against target 1 is non-optimal in this respect. The use of an attack position with maximal effect against target 1 would require a longer tour for both aircraft, and thus this alternative is non-optimal.

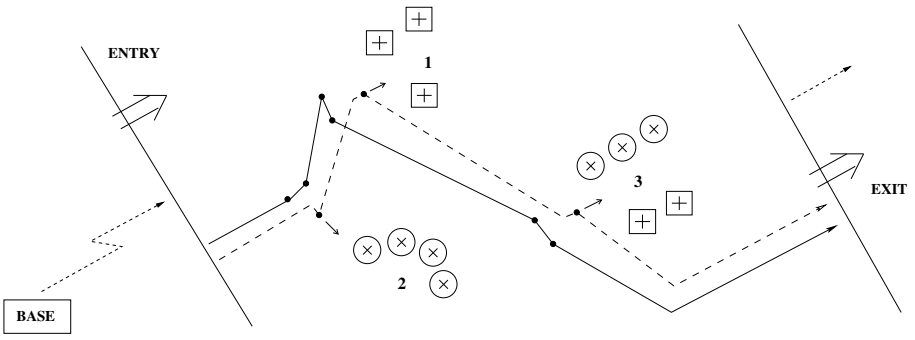


Figure 16: Optimal solution to the example scenario for two aircraft with default settings.

5.2 At most two attacks

With the setting $\Gamma = 2$, a single aircraft is not able to attack all targets. As seen in Figure 17, the new solution uses the same attack and illumination positions as before, with the same attack sequence, but the aircraft switch roles against target 2. The total mission time becomes $t_F = 294$ seconds, and the expected effect of the attacks against each target is the same as before.

After AMPL and CPLEX preprocessing, the problem consists of 4037 rows, 3841 columns and 3723 binary variables, and it was solved in 341 seconds.

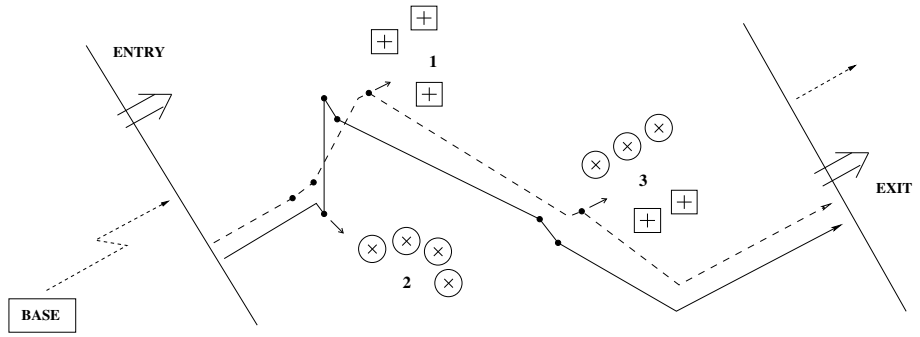


Figure 17: Optimal solution to the example scenario, here for two aircraft with at most two attacks each.

5.3 Precedence

For the third case, we set $\Gamma = 3$ again, but impose precedence constraints stating that target 1 must be attacked before both targets 2 and 3. The result is presented in Figure 18. The attack sequence now becomes 1–2–3, which fulfills the precedence relations. Note that the aircraft switch roles even though not forced to do so.

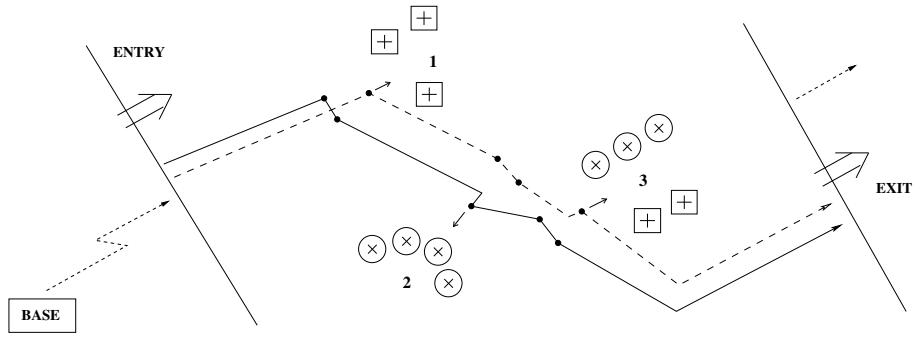


Figure 18: Solution to the example scenario, here with precedence constraints stating that target 1 must be attacked before both targets 2 and 3.

The total mission time becomes $t_F = 295$ seconds, and the expected effect of the attacks against targets 1 and 3 are the same as before, while the expected effect against target 2 is lower than before. Comparing the attack position against target 2 in this solution with the one in previous solutions, one can see that this is due to stronger defense of the SAMs from this direction of attack. After AMPL and CPLEX preprocessing, the problem consists of 3581 rows, 3398 columns and 3278 binary variables, and was solved in 308 seconds.

5.4 Specified roles

For the fourth case, we specify that one aircraft can only perform attacks, and hence the second aircraft can only illuminate targets. The precedence constraints are the same as for the previous case, so that target 1 must be attacked before both targets 2 and 3. The result is presented in Figure 19.

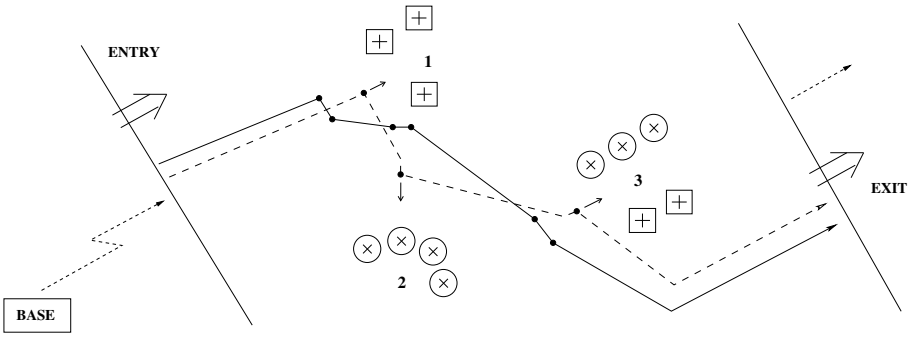


Figure 19: Solution to the example scenario, here with both precedence constraints and specified roles.

The attack sequence is still 1–2–3, but the aircraft do not switch roles anymore. The total mission time is again $t_F = 295$ seconds, and the expected effect of the attacks against each target is the same as in the previous case. Targets 1 and 3 are attacked from the same positions, while target 2 is attacked from another position, when flight paths are optimized with respect to the specified roles. After AMPL and CPLEX preprocessing, the problem consists of 1281 rows, 1158 columns and 1090 binary variables, and was solved in 14 seconds.

5.5 Model comparison and Conclusions

Although the two models presented in Sections 4.2 and 4.5 are mathematically equivalent, their performance can vary significantly. The problem size and runtimes presented for the test case is based on the first model, and in Table 1 we present a comparison between the two models for the same scenario and settings, plus some additional ones.

The number of rows, columns and binary variables after AMPL and CPLEX preprocessing is reported, together with run times (done) and when the optimal solution is found (opt). The given optimal objective value is obviously the same for both models.

From the results of the scenarios presented here, the second model seems superior in every sense, even though the first model finds the optimal solution faster for a few instances. The significantly smaller number of binary variables for the second model clearly has a positive effect on the total run time, as well as the time for finding the optimal solution, and its behaviour on larger instances will be investigated later on in Section 9.7.

Table 1: Model comparison of the two mathematical models. Emphasized numbers are the results for the second model. The columns *rows*, *cols* and *bins* are the result of the AMPL and CPLEX preprocessing. Column *opt* state the time when the optimal solution was found, and column *done* is the total run time.

	AMPL/CPLEX			TIME (s)		
SETTINGS	rows	cols	bins	opt	done	obj
Default	4468	4273	4155	2	359	75.38
$R = 2$	*8463*	*3088*	*140*	*19*	*47*	
Capacity	4037	3841	3723	2	341	75.93
$\Gamma = 2$	*8028*	*3088*	*140*	*3*	*66*	
Precedence	3581	3398	3278	240	308	77.90
`{1|2,3}`	*6696*	*2248*	*128*	*13*	*13*	
Precedence	1281	1158	1090	14	14	77.92
+ S.R.	*2260*	*1208*	*76*	*2*	*2*	
Default	6029	5705	5515	6	628	69.59
$R = 4$	*12307*	*3276*	*244*	*90*	*171*	
Precedence	5688	5357	5153	396	398	71.42
`{1|2,3}`	*10591*	*2507*	*238*	*10*	*81*	
Default	6101	5748	5536	360	743	69.28
$R = 6$	*12420*	*3319*	*265*	*75*	*124*	

Finding an optimal solution through direct application of a general MIP solver to the mathematical model is only practical for smaller scenarios. Even for problem instances including only five targets, it takes CPLEX several hours to verify optimality, although it is able to find feasible and near-optimal solutions much earlier. Hence, efficient heuristics will be needed in order to meet the needs and expectations of real world applications.

In the upcoming Section 7 we focus on a constructive heuristic where the underlying VRP structure is utilized to provide near-optimal solutions, and in Section 8 we outline a column generation inspired approach which exploits the limited number of targets involved in real world scenarios.

6 Underlying Problem Structure

The problem structure of the military aircraft mission planning problem yields a mathematical model that is recognized as a generalized vehicle routing problem with several side constraints. As a foundation for the forthcoming heuristic methods, as they will all utilize the structure of the problem, this section describe an efficient way of transforming the generalized traveling salesman problem for multiple salesmen into the standard traveling salesman problem.

A nice survey of transformations for the multiple salesman problem into the standard traveling salesman problem can be found in [3], together with a description of exact and heuristic procedures for that problem. It is pointed out that transformations to the standard traveling salesman problem results in a highly degenerate problem, which is troublesome when solved in a branch-and-bound setting. The intention here is to solve the traveling salesman problem using a state-of-the-art heuristic, hence this drawback is not crucial.

6.1 The Traveling Salesman Problem

The traveling salesman problem, TSP, is a very well-known problem in optimization. Its first formulation was that a salesman should visit a number of cities, and the question is how he should travel, and in what order he should visit the cities. The objective is to minimize the cost, and the constraints state that each city should be visited exactly once. A nice reference is the book [9].

This problem occurs in many different circumstances, most of them having nothing to do with salesmen. In our case, we can think of an aircraft visiting a number of positions. Efficient solution methods have been developed for the TSP, so if we manage to formulate our problem as a TSP, this may lead to an efficient solution method.

6.2 Generalized multiple TSP

Since both attack and illumination nodes can be seen as clusters of nodes, where exactly one node should be visited in each cluster, we have a Generalized TSP. Such problems are possible to transform into equivalent standard TSP problems though, and the details will be described.

Moreover, if we assume a homogeneous aircraft fleet, the problem becomes a multiple TSP (mTSP), meaning that instead of one salesman we have m salesmen who should visit all nodes in an overall optimal fashion. The mTSP can also be transformed into an equivalent standard TSP.

This special kind of TSP problem, the Generalized TSP for multiple salesmen, will be referred to as the Generalized multiple Traveling Salesmen

Problem (GmTSP). In order to model these problems as ordinary TSP problems, we will utilize and couple the two transformations. For deeper insight and more information on these transformations, we refer to Noon and Bean [10].

The following notation will be used throughout the section. We consider a directed and asymmetric graph $G = (\mathcal{N}, \mathcal{A})$ where each node belongs to one of K clusters, denoted by $C_1, C_2, \ldots, C_K$ and containing $n_1, n_2, \ldots, n_K$ nodes respectively. The graph has in total $N = |\mathcal{N}|$ nodes.

6.3 Generalized TSP into TSP

First consider a TSP where each node belongs to exactly one of K clusters, and exactly one node from each cluster should be visited. Such a Generalized TSP can be transformed into a standard TSP in a few steps.

1. Define a parameter M to be equal to the sum of the costs of the N most expensive arcs.
2. For each cluster C_k, label its nodes arbitrarily by c_{ki}, where $i = 1, \ldots, n_k$. Add zero-cost directed arcs between all nodes so that a cycle is formed, (c_{k,n_k}, c_{k1}) and $(c_{ki}, c_{k,i+1})$ for $i = 1, \ldots, n_k - 1$.
3. Replace each outgoing arc (c_{ki}, c_{lj}), from cluster k to another cluster l, by a new arc $(c_{k,i-1}, c_{lj})$ with arc cost equal to the old arc cost plus M. This is necessary in order to guarantee that all nodes within a cluster are visited consecutively.
4. All other arcs and costs remain unchanged.

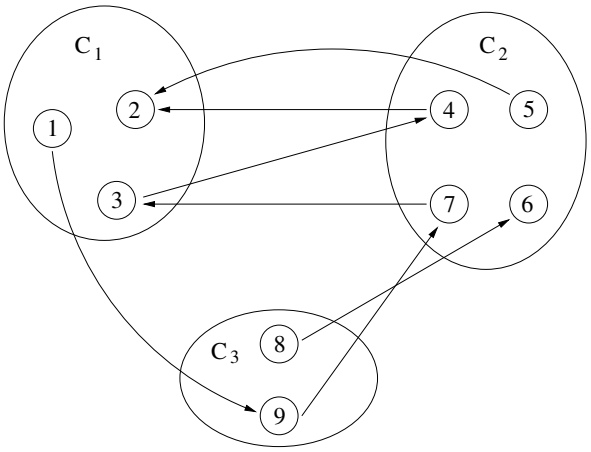
A solution for the new TSP can easily be interpreted in terms of the original problem. Consider the optimal tour and look at an arc entering a cluster. By construction, the following arcs in the tour are all the zero-cost intercluster ones, except the last one as this arc leaves the cluster.

So by only looking at arcs entering and leaving each cluster, it is straightforward to reconstruct the Generalized TSP solution. An example is found in Figure 20.

6.4 mTSP into TSP

An mTSP is a standard TSP problem but with the distinction that the nodes should be visited by m equivalent resources, that is, the nodes should be partitioned between m salesmen whom then should be given a TSP tour each. The mTSP can also be transformed into a standard TSP in a few steps.

Generalized TSP:



Standard TSP:

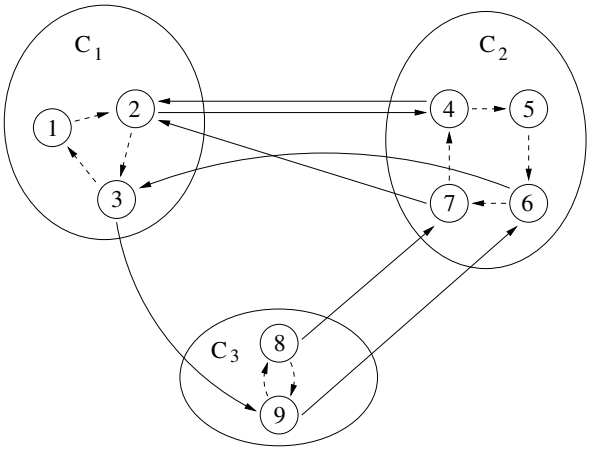


Figure 20: Transformation of a Generalized TSP into a standard TSP. Nodes within a cluster are connected with directed arcs of cost 0, and all outgoing arcs are shifted one step back with respect to the dashed zero cost arcs.

If each salesman is connected to some fixed initialization cost, this can also be handled by the transformation. Assume that the network includes a home city, or depot, labeled node 0, and a set of $n-1$ customer cities.

1. Add dummy nodes, labeled $-1, -2, \ldots, -(m-1)$, one for each additional salesman.
2. Add directed arcs $(-i, j)$ and $(j, -i)$ between all dummy nodes and customers, with the same costs as for arcs $(0, j)$ and $(j, 0)$.
3. Add directed zero cost arcs $(-i, -(i-1))$ for each pair of consecutive dummy nodes.

An example of this transformation is found in Figure 21. Any fixed costs associated with the salesmen are put on the directed zero cost arcs between the dummy salesmen nodes, in increasing order, so that the last salesman is also the most expensive one.

The last step is optional, and if omitted, then all m salesmen are forced to leave the depot. Using a single salesman is usually the most cost efficient way to visit all customers, and thus the point of having multiple salesmen is not utilized. Hence it is useful to be able to force all salesmen to participate.

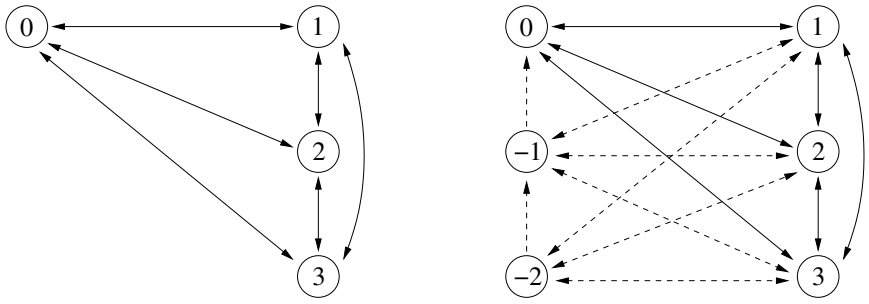


Figure 21: Transformation of an mTSP into standard TSP. One dummy node for each extra salesman is added, plus some new arcs.

A solution for this new TSP can easily be interpreted in terms of the original problem. Without any loss of generality, assume the optimal tour starts at node 0. As soon as the tour visits one of the dummy nodes, this is interpreted as the current salesman is done and the next one starts his tour.

For example, the tour `0-2-3-(-2)-(-1)-1-0` in the transformed problem above should be interpreted as the first salesman goes from the depot to visit node 2 and then node 3, and the second salesman goes from the depot to node 1 and then back. The third salesman is not used.

6.5 GmTSP into TSP

We now put the two transformations together, hence giving a transformation from a Generalized mTSP into a standard TSP problem.

In the first transformation, from Generalized TSP into TSP, the directed cycles between nodes inside each cluster are added, a total of N extra arcs. Besides this, the only change is the coupling of arcs between already existing nodes. Assuming that nodes within each cluster are numbered consecutively, this recoupling can be seen as a simple shift of rows. Consider a Generalized TSP problem consisting of K clusters, each containing $n_1, n_2, \ldots, n_K$ nodes, and N nodes in total. The node-node adjacency matrix for this problem is described in a $N \times N$ block matrix A with empty main diagonal blocks, since there are no intercluster arcs.

$$A = \begin{bmatrix} & A_{12} & A_{13} & \ldots & A_{1K} \\ A_{21} & & A_{23} & \ldots & A_{2K} \\ A_{31} & A_{32} & & \ddots & \vdots \\ \vdots & \vdots & \ddots & & A_{K-1,K} \\ A_{N1} & A_{K2} & \ldots & A_{K,K-1} & \end{bmatrix}$$

Consider also a block permutation matrix P, where each block matrix P_k has the same structure but depends on n_k, the size of the cluster. Let I_n denote an identity matrix of size $n \times n$, then

$$P = \begin{bmatrix} P_1 & & & \\ & P_2 & & \\ & & \ddots & \\ & & & P_K \end{bmatrix} \quad , \quad P_k = \begin{bmatrix} 0 & I_{n_k-1} \\ 1 & 0 \end{bmatrix}$$

Let $\tilde{A}_{kl} = P_{n_k} \cdot A_{kl}$ denote the rowshift for arcs leaving cluster k. The transformed problem can now be described as

$$\tilde{A} = \begin{bmatrix} P_{n_1} & \tilde{A}_{12} & \tilde{A}_{13} & \ldots & \tilde{A}_{1K} \\ \tilde{A}_{21} & P_{n_2} & \tilde{A}_{23} & \ldots & \tilde{A}_{2K} \\ \tilde{A}_{31} & \tilde{A}_{32} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & P_{n_{K-1}} & \tilde{A}_{K-1,K} \\ \tilde{A}_{K1} & \tilde{A}_{K2} & \ldots & \tilde{A}_{K,K-1} & P_{n_K} \end{bmatrix} = P(A + I_N).$$

Hence the network structure for the transformed problem is found by adding an identity matrix to the main diagonal and then multiply by the permutation matrix P, with the structure described above.

For the second transformation, assume that matrix $\tilde{A}$, still of size $N \times N$, does not include origin or destination nodes. For an instance with m salesmen, we add m extra nodes (new rows and columns) with directed arcs to and from all previous nodes, that is $m(2N+1)$ new arcs. The new adjacency matrix $\hat{A}$ now include arcs to and from the origin and destination nodes, described by matrix E of size $N \times m$ with all entries equal to 1.

Matrix B_m corresponds to the extra arcs between the added dummy nodes.

$$\hat{A} = \begin{bmatrix} \tilde{A} & E \\ E^T & B_m \end{bmatrix}, \quad B_m = \begin{bmatrix} 0 & 0 \\ I_{m-1} & 0 \end{bmatrix}$$

In the case of forcing all salesmen to be active, matrix B_m consists of all zeros instead. The structure makes it very easy to transform the GmTSP into a standard TSP, with a moderate m extra nodes and $2mN + m + N$ extra arcs.

6.6 The LKH solver

The TSP problem is well studied, and although NP-hard there exist successful heuristic approaches. One of the best available solvers is the LKH solver, an implementation of the Lin-Kernighan heuristic for solving the traveling salesman problem, presented in their paper [14] from 1973. The heuristic involves swapping pairs of sub-tours to make a new tour.

The algorithm is a generalization of 2-opt and 3-opt, local search strategies that work by switching two or three paths to make a given tour shorter. Lin-Kernighan is adaptive and at each step decides how many paths between cities need to be switched to find a shorter tour. The implementation of LKH has been made in the programming language C by Keld Helsgaun. We recommend [7] for more information and details on the implementation.

7 Constructive Heuristics

A great advantage with this problem is that, even though it is very hard to solve, feasible solutions can be found rather easily. In this section, we present a constructive greedy heuristic, which consist of four steps. Each step will be described in detail in the upcoming sections, but a short overview is given here to outline the heuristic.

A heuristic solution will be denoted $(\bar{x}, \bar{t})$, where $\bar{x}$ is short for the routing variables x^r_{ij}, and $\bar{t}$ represents the continuous time variables t^A_m, t^I_m, t^r_i, and t_F. Assume a fleet of $2m$ aircraft, working in pairs, that is m resource pairs.

1. Solve a GmTSP for m aircraft over clusters of attack nodes for each target, in order to partition the targets and decide an attack sequence.
2. Solve m shortest path problems, one for each resource pair, in order to construct a feasible routing solution $\bar{x}$.
3. From step 1 and 2, we have created a partial solution in $\bar{x}$, and now need to find $\bar{t}$. To do this, solve a Project Network problem.
4. Optionally, perform a local search on $(\bar{x}, \bar{t})$.

7.1 Assumptions

For this heuristic to work, we need to introduce some assumptions and limitations. The mathematical models presented in Section 4 are able to handle a heterogeneous aircraft fleet, but in order to use the transformations described in Section 6 we must restrict ourselves to a homogeneous aircraft fleet. Further, if a problem instance contains precedence constraints, the target sequence must be either partially or totally ordered. This limitation is acceptable for any real case scenario, where wind conditions and other factors will implicate an ordering, and hence is not an issue in practice.

Any precedence relations that fulfill this assumption are handled in step 1, where the GmTSP network is adjusted by removing all arcs violating any precedence relation. Hence each subproblem in step 2 will generate feasible paths, and in step 3 we add dummy arcs to the project network in order to force the critical path to satisfy all precedence relations.

To illustrate the necessity of this limitation, we present a small example in Figure 22 where the only precedence constraint is that target 2 must be attacked before target 3. The solution from step 1 could possibly violate this, and hence the solution will not be feasible. For any partially ordered set of targets, this cannot happen.

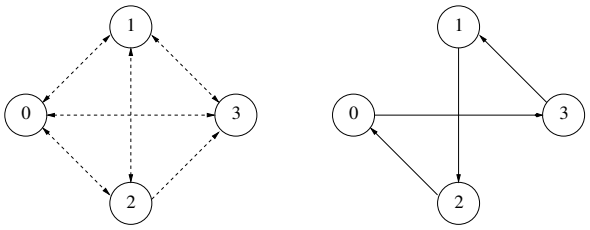


Figure 22: The aircraft should start and stop at node 0, and precedence states that target 2 must be attacked before target 3. Even so, a feasible path in this network is 0–3–1–2–0, which contradicts the precedence constraint.

At the moment, we also assume that the aircraft fleet have no armament capacity constraints, that is, each aircraft is allowed to perform an unlimited number of attacks.

7.2 Step 1 and 2 - GmTSP and Shortest Path

Under the assumption that all resources work in pairs, it is possible to construct GmTSP problems and utilize the method presented in Section 6.2 to solve them. Hence, we assume an aircraft fleet of size $2m$. In the first step, we construct clusters which contain only attack nodes and set up a GmTSP for m aircraft, where arcs are valid movements between these nodes like before. The solution to this GmTSP provides a partitioning of all targets between the aircraft pairs, specifies an attack sequence of the targets and which attack nodes to be used.

In the second step we solve m small shortest path problems, one for each aircraft pair, where each network is directed and acyclic, and consist of illumination nodes that are compatible with the corresponding attack nodes chosen by step 1.

The result is probably suboptimal, since the path of the illuminator is likely to be more time consuming than the path of the attacker due to the fact that illumination is performed for 15 seconds while an attack is performed instantaneously. A small example is found in Figure 23.

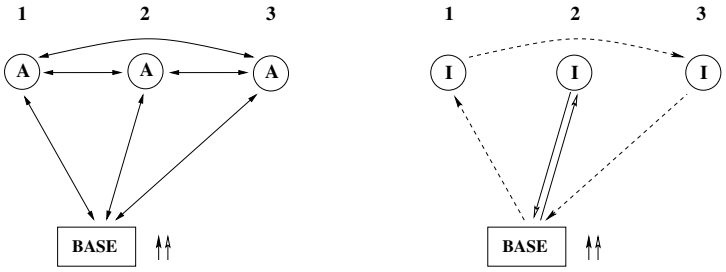


Figure 23: An example with 3 targets and 2 aircraft pairs. In step 1, to the left, an mTSP with attack nodes only. In step 2, to the right, one should solve a Shortest Path problem for each aircraft pair, and find compatible illumination nodes that match the solution from step 1.

7.2.1 Specified Roles

Instead of limiting the aircraft fleet to work in fixed pairs, assume we specify m aircraft to be attackers and k aircraft to be illuminators. Step 1 can still be solved as a GmTSP for clusters of attack nodes, but step 2 need some adjustment. The solution from step 1 provides a partial ordering of the attacks against each target, which needs to be satisfied when routing the k illuminating aircraft, as well as the given precedence relations.

The subproblem is no longer a shortest path problem, instead we need to set up and solve a second GmTSP. In the case of no given precedence relations, it is sufficient to consider the partial ordering of step 1 when defining the underlying network for the GmTSP. This can be done by ordering the targets with respect to earliest visit time of the attackers. One can now solve the new GmTSP problem for the k illuminators, on a network with illumination nodes compatible with the specified attack nodes from step 1, and directed arcs not violating the fixed sequence. In the case of a totally ordered attack sequence, the GmTSP problem can be solved directly.

With a partially ordered attack sequence, it is necessary to set up a Project Network (see next section) and find a topological ordering of the targets. With this ordering at hand, the GmTSP problem can be solved for the k illuminators as before.

7.3 Step 3 - Project Network

So far we have a partial solution, as the first two steps have created a feasible routing of our resources, i.e. the x_{ij}^r variables. The third step consists of creating feasible values for our time variables t_i^r, t_m^A, t_m^I and t_F, matching the routing. This can be done in an elegant way, by constructing a Project Network and find its critical path.

Let each target be represented by an activity node in this network, and add directed arcs wherever at least one resource goes directly from one target to another. If resources work in pairs, they will obviously visit their targets in the same sequence. Instead of adding two parallel arcs, it is sufficient to add the one with longest traveling time, i.e. biggest T_{ij} value. Precedence relations are handled by adding dummy arcs, one for each pair $(m, n) \in \mathcal{S}$.

By sorting the nodes in a topological order, the project network becomes a Directed Acyclic Graph (DAG). The critical path can then be found directly by the Bellman equation. Node prices for each target node will correspond to the t_m^A and t_m^I variables, and the total mission time, t_F, is the cost of the critical path.

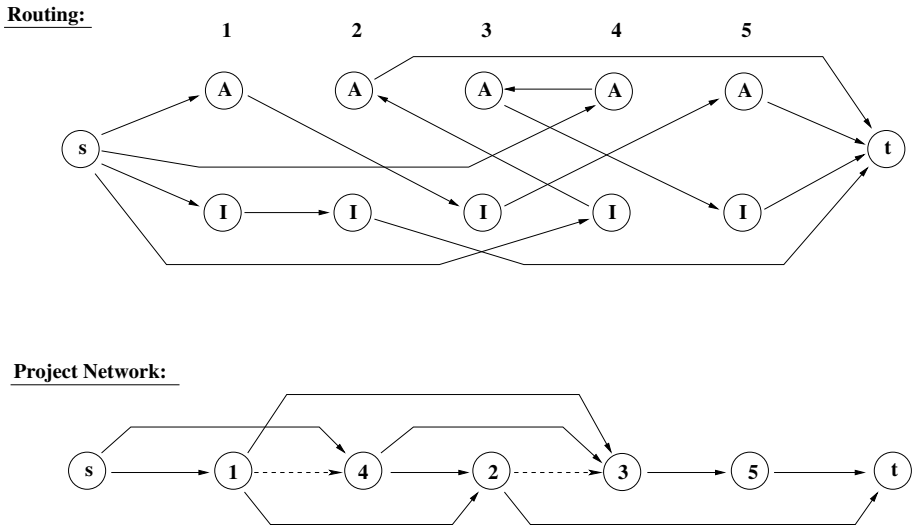


Figure 24: Example with 5 targets and 4 aircraft. The upper picture shows the flight paths of each aircraft. The lower picture is the corresponding project network in which to find the critical path. The dashed lines correspond to dummy arcs, used to impose precedence relations. Here target 1 must be attacked before target 4, and target 2 before target 3.

An example from a more general setting is found in Figure 24, including four aircraft that are not forced to work in pairs, and five targets with given precedence relations $(4, 1)$ and $(3, 2)$. The project network problem will find the optimal values for the time variables, even for the more general setting in this example, as long as the network it is acyclic.

7.4 Step 4 - Local Search

In a heuristic framework, a local search procedure can often be used to improve the solution. For this problem, there exist a natural neighbourhood involving the attack and illumination combinations, (A/I)-combinations, for the active sectors in a given feasible solution $(\bar{x}, \bar{t})$. We define a neighbour solution as any solution where the sectors are kept, but at least one attack or illumination node is changed.

As mentioned earlier, the path for the second aircraft in each pair is likely to become more time consuming, since the first step is solved in a greedy manner. The local search will try to compensate for this, as it will test solutions where the roles of each aircraft against a target is switched. Assume there are n_A attack nodes and n_I illumination nodes in each sector. Then there exist $n_A n_I$ combinations, or neighbours, given that the aircraft do not switch roles. With the possibility of changing attacker and illuminator against a target, a factor 2 is added and there are $2n_A n_I$ combinations in each active sector.

Our local search evaluates all (A/I)-combinations for the active sectors, and a problem with M targets will have M active sectors, hence $M \cdot (2n_A n_I)$ neighbouring solutions where exactly one sector at a time is changed.

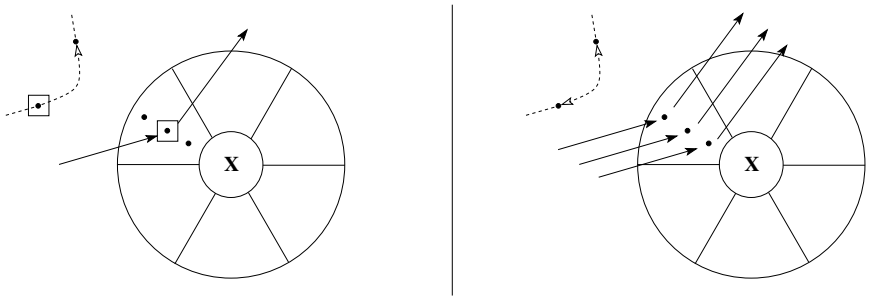


Figure 25: Local Search Procedure. To the left, the active sector and the currently used attack and illumination nodes are marked by a box. To the right, an illustration of the $2 \cdot 3 = 6$ combinations of attack and illumination nodes to be evaluated. If the aircraft can switch roles, there are 12 combinations.

For a wider search, more sectors can be changed simultaneously, and we extend the neighbourhood to involve more than one change at a time. The neighbourhood of a solution $\bar{x}$, allowing k sectors to change simultaneously, is denoted $\mathcal{N}_k(\bar{x})$.

The size of neighbourhood $\mathcal{N}_k(\bar{x})$, i.e. the number of combinations when k out of M sectors are changed at once, is $\binom{M}{k} \cdot (2n_A n_I)^k$ with the extreme case of $(2n_A n_I)^M$ combinations for $\mathcal{N}_M(\bar{x})$. Since the current combination is already evaluated, one combination can always be removed.

Each such combination corresponds to a new solution $\bar{x}$, and hence the $\bar{t}$ variables needs to be recalculated by the critical path routine described in the previous section.

7.5 Extensions and Future Work

In this section, we present some alternative versions of the constructive heuristic, together with some possible extensions and future work. To start with, it is not obvious what arc costs should be used when solving the subproblems, and some alternative versions are discussed. Further, in the first step of the heuristic, it is not necessary to consider only clusters of attack nodes. Also, the heuristic cannot handle the armament constraints at the moment, and a possible solution to this limitation is discussed.

Arc costs

It is not obvious what the arc costs should be when defining and solving the GmTSP problem in step 1, and the shortest path problems in step 2. When routing attackers, it seems natural to consider expected effect against the targets, hence using the c_{ij}^r values. In this way, one can avoid finding solutions with low expected attack efficiency, caused by focusing on minimizing time consumption.

On the other hand, if minimizing the mission time span is of great importance, it is better to use T_{ij}^r as arc costs. This will probably result in solutions with poor expected attack efficiency though. One could of course consider only "good enough" attack positions to start with, hence solving this issue in advance.

Another way to handle these conflicting goals is to specify weight parameters a and b for c_{ij}^r and T_{ij}^r respectively, and hence solve the problem with arc costs $w_{ij}^r = a \cdot c_{ij}^r + b \cdot T_{ij}^r$. A possible extension is to create Pareto-optimal solutions by experimenting with the values of a and b, and this is ongoing work at the moment.

Clusters

In the first step, it is possible to construct clusters with illumination nodes instead of attack nodes for each target, emphasizing the fact that illumination is more time consuming than an attack. The second step then involves compatible attack nodes instead, and can be solved as before.

A third option is to consider one big cluster of nodes for each target, containing both attack and illumination nodes. The solution of the first step will still consist of one flight path for each aircraft pair, but with the possibility of visiting both attack and illumination nodes. The second step is the same as before, but now with clusters of attack or illumination nodes for each target, depending on the solution from the first step. In all, both these options essentially give rise to the same subproblems as before, just with different node sets. The most natural option is to consider attack nodes though, since the objective is primarily to maximize the expected effect against.

There is an obvious side effect of considering clusters containing both attack nodes and illumination nodes in step 1. As pointed out earlier, the paths from step 2 are probably more time consuming, since the flight paths from step 1 are optimized with respect to the first aircraft in each pair. When more alternatives are available in the first step, the route of the first aircraft could become even tighter in time, and the compatible flight paths found in step 2 probably give rise to waiting times, and the overall solution is likely to become suboptimal.

Armament Constraints

It is possible to extend the method to handle the armament constraint, that each aircraft can attack at most Γ targets, but it is not straightforward. The problem is that the GmTSP solutions from step 1 might generate paths where an aircraft visits more than Γ targets, and it is not possible to incorporate such constraints into the network formulation.

With an aircraft fleet of size $r \geq \lceil M/\Gamma \rceil$, this issue can be solved by considering a partitioning of the targets into r subsets, where each subset consist of at most Γ targets. By solving one Generalized TSP for each such subset of targets, and one shortest path problem for the accompanying illuminator, the best partitioning will yield a solution that is feasible with respect to the armament constraints.

The drawback is that the Constructive Heuristic needs to be applied to all subsets in order to compare their solutions. As an example, for three targets, $\Gamma = 2$ and $r = 2$, there are three feasible partitionings:

```
{1,2|3}, {1,3|2}, {2,3|1}
```

The number of feasible partitionings increase quickly though. For five targets, $r = 3$ and $\Gamma = 2$, there are 15 feasible partitionings.

8 Column Enumeration

Optimization problems can often be formulated in different ways, thereby enabling different solution approaches. One quite special way is to use a so called "column" formulation. The basic idea is that a solution is a combination of several simpler types of solutions, for example, a solution may consist of several traveling salesman tours combined in some way.

Then one might formulate the original problem so that each of the simpler solutions is represented by a column in the "master" problem, and the task of the master problem is to find the best combination of these columns. This leads to a master problem of set covering or set partitioning type.

Column Generation

In the solution approach called *column generation*, one starts with a small set of columns for which the master problem is solved. With the help of information from this solution, like the dual solutions and reduced costs, one formulates an optimization problem for generating the most promising column not yet included.

A somewhat simpler solution approach is to generate all possible columns, without the use of an optimization problem which may be difficult to formulate, and instead keep all columns in a pool. The master problem then operates on a subset of columns, since it would be too difficult to solve if all these columns were included, and then sequentially add columns that are needed. It is also possible to exclude columns from the master problem in order to keep down the size of the subset.

Columns for our problem

The problem size, with respect to the number of variables (nodes and arcs), depends on the discretization of feasible attack positions and compatible illumination positions, and increases exponentially. One thing that does not depend on the discretization, and is very limited, is the number of targets.

With $M = |\mathcal{M}|$ targets, there exists $K = 2^M - 1$ nonempty subsets S_k of targets. A large problem scenario includes no more than 10 targets, hence in our worst case there are $K = 1023$ subsets. Let $\mathcal{K}$ denote the set of all subsets S_k for a given $\mathcal{M}$.

For each subset S_k, we find the optimal feasible path for one aircraft pair, that is a sequence for the targets in S_k and which nodes each aircraft in the pair should visit. Such a solution, also referred to as a column, visits all targets in S_k and provides a mission time t_k and an expected attack value c_k.

With an optimal solution for each subset in $\mathcal{K}$ it only remains to decide which columns to use in order to cover each target exactly once. In other words, the master problem becomes a Set Partitioning Problem (SPP) with K sets.

8.1 Generate Columns

In order to solve the SPP, we need to generate columns, that is to find an optimal solution for each subset S_k. We will formulate and solve shortest path subproblems for one aircraft pair, and utilize the fact that a large scenario includes at most ten targets. For a subset S_k and one aircraft pair, the targets must be visited in the same sequence. Since each target is either attacked or illuminated by the first aircraft, the actions of the second aircraft follows as a direct consequence. One must still make sure that the attack and illumination are performed in the same sector though.

Assume that a target sequence for S_k is specified. For a given sector, and a specified role of each aircraft for each target, the subproblem becomes a shortest path problem on a small directed acyclic network for each aircraft, which is extremely easy to solve. Even though there are many such combinations of sequence, sector and role, we propose to test all of them due to the limited number of targets. Since a large problem include at most $M = 10$ targets, the number of combinations are kept to a manageable size.

Combinations

For a given subset S_k, with $m = |S_k|$ targets, there are at most $m!$ target sequences. Any precedence relations stated in $\mathcal{S}$ will reduce this number, and in the extreme case of a totally ordered sequence there is exactly one combination. For a partially ordered set, where n_1 targets should be attacked before n_2 targets, the number of combination becomes $n_1! \cdot n_2!$.

For m targets, each with G sectors, there is G^m sector combinations. This is also an upper bound since all sectors does not have to be in use, for example due to protected objects in the vicinity. More generally, let g_i be the number of sectors in use for each target $i \in S_k$, then there are $\prod_{i \in S_k} g_i$ sector combinations.

If no restrictions are imposed on the aircraft pair, the number of Attack and Illumination (A/I) combinations against m targets is given by 2^{m-1}, as each target is either attacked or illuminated by one aircraft and vice versa for the other aircraft. If each aircraft has a specified role, either attacker or illuminator, there is obviously only one A/I combination.

Under the assumption that each aircraft can perform at most Γ attacks, an aircraft pair can visit at most 2Γ targets. For a value of $\Gamma = 2$, this limits the size of feasible subsets to include $m \leq 4$ targets. It also limits the number of A/I combinations for an aircraft pair. For a subset with one or two targets, this restriction does not affect the number of combinations, but for a subset with three or four targets there are only three valid combinations: `A-A-I`, `A-I-A`, `A-I-I` for three targets, and `A-A-I-I`, `A-I-A-I`, `A-I-I-A` for four targets, instead of $2^{3-1} = 4$ and $2^{4-1} = 8$ combinations.

Table 2 states the number of combinations of sectors, target sequences and (A/I) combinations, that is the total number of subproblems to be solved for a subset of given size. The different assumptions for (A/I) combinations are No Assumption (NA), at most two attacks (M2) and Specified Roles (SR). Here we use $G = 6$ sectors for each target, and the number stated in the rightmost column is an upper bound on the total number of subproblems to be solved based on the M2 assumption. Under the assumption of at most two attacks, it is only possible to consider subsets S_k with at most four targets. As seen above, the number of combinations explode even for such relatively small sets. For the case of no assumption, this number is even larger, and for the specified roles assumption the number is decreased significantly.

Table 2: The total number of combinations to be tested in order to find the optimal route for one aircraft pair against the m targets in subset S_k.

# Targets in subset	# A/I Combinations			# Sector Combinations	# Target Sequences	Total # Subproblems
$m = \|S_k\|$	NA	SR	M2	G^m	$m!$	For M2
1	1	1	1	6	1	6
2	2	1	2	36	2	144
3	4	1	3	216	6	3888
4	8	1	3	1296	24	93312
5	16	1	-	7776	120	-

Subproblems

We have now investigated how many combinations of A/I, sectors, and target sequences that arise for a given subset of targets. Assume now that we have a problem with M targets and that each aircraft can attack at most two targets, how many subproblems do we need to solve in order to find the optimal columns? In general, there are $\binom{M}{m}$ subsets with m targets, so for example with $M = 4$ targets the following subsets exist; 4 including one target, 6 including two targets, 4 including three targets and obviously only 1 set including all four targets. For each such subset, there is a certain number of subproblems to be solved, as stated in Table 2.

In Table 3, the total number of subsets and subproblems for a given problem size is presented, based on the assumption of at most two attacks. As an example, for $M = 4$ targets the total number of subproblems to be solved in order to find the optimal columns is calculated to be $4 \cdot 6 + 6 \cdot 144 + 4 \cdot 3888 + 1 \cdot 93312 = 109752$. Once again, this is an upper bound on the number of subproblems to be solved, and depends on the precedence constraints for the problem instance at hand.

The assumption that an aircraft can attack at most two targets limits the subsets to include at most four targets, but even if the aircraft could perform more attacks, it is still reasonable to only consider subsets of this size. For example, suppose a problem includes 6 targets and 2 aircraft pairs. In theory one pair could engage five of the targets and leave one target to the other pair, but it seems more reasonable that an optimal solution would partition the targets more evenly between the aircraft pairs.

As seen in Table 3, the number of subproblems to solve increase quickly. It must be stated that each subproblem is extremely small and trivial to solve, as seen in next section, and the process of solving these subproblems is fully parallelizable. With enough computer resources, the columns can be found in reasonable time.

Table 3: The total number of subproblems to be solved in order to find the optimal columns for a problem with M targets. These numbers are based on a discretization of $G = 6$ sectors, and the M2 assumption.

# Targets	# Subsets of size 1–4				# Subproblems	# Columns
$M = \lvert\mathcal{M}\rvert$	$\binom{M}{1}$	$\binom{M}{2}$	$\binom{M}{3}$	$\binom{M}{4}$	TOTAL	$2^M - 1$
1	1	-	-	-	6	1
2	2	1	-	-	156	3
3	3	3	1	-	4 338	7
4	4	6	4	1	109 752	15
5	5	10	10	5	506 910	31
6	6	15	20	15	1 479 636	63
7	7	21	35	35	3 405 066	127
8	8	28	56	70	6 753 648	255
9	9	36	84	126	12 089 142	511
10	10	45	120	210	20 068 620	1023

8.2 Solving subproblems

Although there are many combinations, and hence many subproblems, each of them is extremely easy to solve. For specified sectors, target sequence and specified roles for each aircraft against each target, the resulting subproblem is a Shortest Path Problem on a directed acyclic graph. Such problems can be solved directly by Bellman's equations.

When the optimal arcs are known, the time of attack and time of illumination against each target is found by solving a Project Network Problem. But since the network is decoupled for the two aircraft, the solution is simply found as the longest arcs between each target.

Notice that these subproblems are solved very efficiently independently of the number of attack and illumination positions in each sector. Hence a much denser discretization of the feasible attack space will not affect the complexity of these subproblems. An increased number of sectors, G, will on the other hand have an immediate effect on the efficiency of this approach.

Each subproblem solution contains a flight path for each aircraft, it specifies which nodes to visit and in what order, a mission value c_k and an overall mission time t_k. When all subproblems have been solved for a subset S_k, these parameters correspond to the optimal flight paths with respect to arc costs c_{ij} for one aircraft pair. If the time aspect is just as important as the mission value, the subproblems can be solved with arc costs T_{ij} instead of c_{ij}, generating time efficient paths.

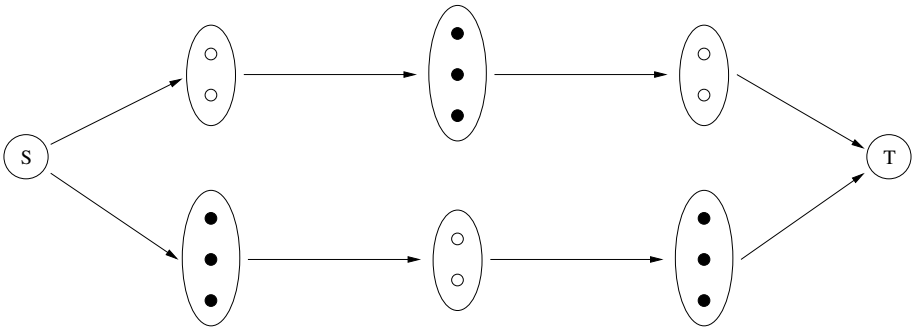


Figure 26: Each subproblem is a Shortest Path Problem on a directed acyclic graph. In this example, one aircraft should perform the following tasks: Attack–Illuminate–Attack. The other aircraft will obviously perform the opposite action at each target.

The use of T_{ij} as arc costs may however result in solutions with low expected attack efficiency, and one should find a reasonable trade off by weighting the two costs together, similar to the ideas presented in Section 7.5.

Once the optimal columns have been found for all subsets in $\mathcal{K}$, we are ready to formulate the Master Problem, a Set Partitioning Problem where all targets should be visited exactly once by the available aircraft fleet.

Greedy column generation

The number of subproblems to be solved grow rapidly with the number of targets, and becomes significant even for instances with as few as five or six targets. To keep the method tractable for all problem sizes, one could generate columns for each subset S_k in a greedy manner instead of evaluating all combinations, and we propose to use the constructive heuristics from Section 7. By doing so, the columns are probably suboptimal, but can be found within seconds instead of possibly hours. This is important if computer resources are limited. A possible extension would be to generate Pareto-optimal solutions by using weighted arc costs as described in Section 7.5.

8.3 Master Problem

For a given set of subtours, $\mathcal{K}$, each covering a subset of targets, we define parameter a_{km} to be one if target $m \in \mathcal{M}$ is covered by subtour $k \in \mathcal{K}$. Each subtour k has a value of c_k, mission time t_k and require n_k resources. We introduce the binary decision variables z_k, one if subtour k should be used, and time variable t_F denotes the total mission time, i.e. when all resources are back. With R resources available, the optimization problem becomes

$$
\begin{aligned}
\max \sum_{k\in\mathcal{K}} \quad & c_k z_k - \mu t_F & & & [MASTER] \\
s.t. \sum_{k\in\mathcal{K}} \quad & a_{km} z_k &= \quad 1 & \quad m \in \mathcal{M} & (1) \\
\sum_{k\in\mathcal{K}} \quad & n_k z_k &\leq \quad R & & (2) \\
& t_k z_k &\leq \quad t_F & \quad k \in \mathcal{K} & (3) \\
& z_k &\in \quad \{0,1\} & \quad k \in \mathcal{K} & (4) \\
& t_F &\geq \quad 0 & & (5)
\end{aligned}
$$

which is easily solved by a general MIP solver like CPLEX. Constraint (1) states that each target should be part of exactly one subtour, and constraint (2) makes sure that the required number of resources do not exceed the available fleet size. Constraint (3) is used to calculate the total mission time.

With the optimal solution z^* at hand, it is straightforward to construct the corresponding $(\bar{x}, \bar{t})$ solution by using the indicated columns. If no precedence relations are given, the achieved solution from the master problem is optimal with respect to the generated columns, and the value of t_F is correct.

Validate solution

The solution of the master problem is always feasible with respect to routing constraints. However, if there exist precedence relations, the mission times t_k for each separate subtour $k \in \mathcal{K}$ might not be realizable since these flight paths will probably affect each other, as illustrated in Figure 27 on next page.

In this example, three aircraft pairs are considered, and the solution of the master problem consist of three subtours: $s-1-4-t$ of length 35, $s-2-3-t$ of length 45, and $s-5-t$ of length 25. The solution states that $t_F = 45$, the longest of the three subtours. However, with precedence relations stating that target 1 must be attacked before targets 2 and 3, which in their turn must be attacked before targets 4 and 5, this is not realizable anymore. Notice that each of the three subtours satisfy these constraints, but from the corresponding project network, found in Figure 27, the length of the critical path states that $t_F = 56$.

Hence the master problem always provide a feasible solution, but likely with an optimistic value for the total mission time. If the time aspect is only secondary, this poses no problem since the paths are still valid. It is possible to solve this problem more accurately, by considering travel times and precedence relations in the master problem. Such an extension of the master problem is presented in next section.

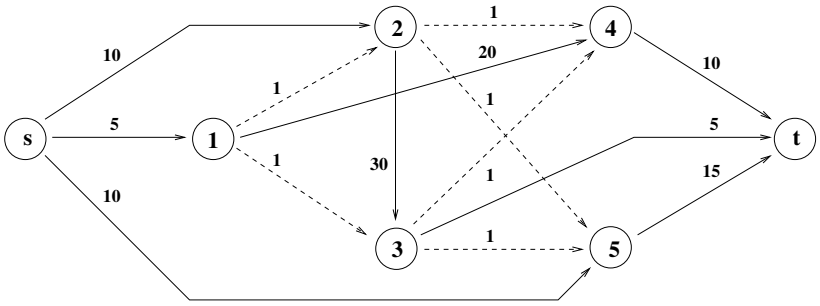


Figure 27: The project network corresponding to three subtours. Due to the given precedence relations, here indicated with dashed arrows, the critical path becomes $s-2-3-5-t$ with a total length of 56.

Pareto-optimal columns

If a solution, when validated, turns out to be extremely time inefficient due to the precedence constraints, it is possible to handle this by iteratively adding constraints that forbid certain column combinations in the Master Problem. For example, if the MP solution suggests column k and column l to be used, we add the constraint $z_k + z_l \leq 1$ and solve it again. This strategy is reasonable since the solution time of the master problem is neglectable.

The proposed approach is very crude though, since forbidding the combination of two columns actually means forbidding that particular partitioning of the targets between the two aircraft pairs. The reason for that combination of columns to result in a bad solution is due to the sequence of the targets, not its partitioning. To add some flexibility, we propose to save the best column with respect to each ordering of targets within each subset S_k, instead of only the best one. This implies no additional work since all combinations are tested anyway. For the Greedy column generation approach, instead of one GmTSP for each subset, one Shortest Path problem needs to be solved for each valid ordering. The process of adding constraints for pairwise columns could be repeated until no better solution, with respect to the validated solution, is found.

8.4 Extended Master Problem

In order to fully capture the precedence relations, as done in the full model, one must consider the individual travel times for each arc included in the subtours. From the generated subtours, a reduced network is constructed.

Let $\widetilde{N}$ denote all nodes in the reduced graph, including dummy origin and destination nodes. Each target $m \in \mathcal{M}$ is associated with attack nodes $\widetilde{N}_m^A$ and compatible illumination nodes $\widetilde{N}_m^I$. Let $\widetilde{A}$ denote all arcs in the reduced network, and $\widetilde{T}_{ij}$ the minimum time needed to traverse arc $(i,j) \in \widetilde{A}$.

Most of the notation is kept from the basic MP, like the set of subtours $\mathcal{K}$, variables z_k and t_F, and parameters a_{km}, c_k and n_k. We introduce the binary arc variables x_{ij} and time variables t_i, t_m^A and t_m^I like in the full model.

Arc variable x_{ij} is equal to one if arc $(i,j) \in \widetilde{A}$ is part of any chosen subtour, and zero otherwise. The continuous time variable t_i is the time at which some aircraft visits node $i \in \widetilde{N}$ and is zero if the node is not part of any chosen subtour. Variables t_m^A and t_m^I are the time of attack and illumination, respectively, of each target $m \in \mathcal{M}$.

Let p_k specify which arcs that are part of the n_k paths defined by subtour k. Define set $K_{ij} = \{k : (i,j) \in p_k\}$ for each arc $(i,j) \in \widetilde{A}$ to describe which subtours k that include arc (i,j). As before, let $\mathcal{S}$ denote the set of ordered pairs (m,n) of targets such that target m cannot be attacked before target n. If no precedence relations are given a priori, set $\mathcal{S}$ is empty. The Extended Master Problem becomes:

$$\max \sum_{k\in\mathcal{K}} c_k z_k - \mu t_F \qquad [Extended\ MASTER]$$

$$s.t. \quad \sum_{k\in\mathcal{K}} a_{km} z_k = 1 \qquad m \in \mathcal{M} \qquad (1)$$

$$\sum_{k\in\mathcal{K}} n_k z_k \leq R \qquad (2)$$

$$\sum_{k\in K_{ij}} z_k = x_{ij} \qquad (i,j) \in \widetilde{A} \qquad (3)$$

$$t_i + \widetilde{T}_{ij} x_{ij} \leq t_j + T_{max}(1 - x_{ij}) \qquad (i,j) \in \widetilde{A} \qquad (4)$$

$$t_i \leq T_{max} \sum_{(i,j)\in\widetilde{A}} x_{ij} \qquad i \in \widetilde{N} \qquad (5)$$

$$t_o = 0 \qquad (6)$$

$$\sum_{i\in\widetilde{N}_m^A} t_i = t_m^A \qquad m \in \mathcal{M} \qquad (7)$$

$$\sum_{i\in\widetilde{N}_m^I} t_i = t_m^I \qquad m \in \mathcal{M} \qquad (8)$$

$$t_m^A = t_m^I \qquad m \in \mathcal{M} \qquad (9)$$

$$t_m^A \geq t_n^A \qquad (m,n) \in \mathcal{S} \qquad (10)$$

$$t_F \geq t_d \qquad (11)$$

$$z_k \in \{0,1\} \qquad k \in \mathcal{K} \qquad (12)$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in \widetilde{A} \qquad (13)$$

$$t_i,\ f_F \geq 0 \qquad i \in \widetilde{N} \qquad (14)$$

$$t_m^A,\ t_m^I \geq 0 \qquad m \in \mathcal{M} \qquad (15)$$

Like before, constraint (1) states that each target should be part of exactly one subtour, and constraint (2) makes sure that the required number of aircraft do not exceed the available fleet size. Constraint (3) couples the z_k and x_{ij} variables, making sure that only arcs that belong to active tours are utilized.

Constraint (4) propagates time, constraint (5) make sure that $t_i = 0$ if node i is not visited, and constraint (6) set the mission starting time to zero. Constraints (7)–(9) define the time of attack and illumination against each target, and make sure that these times are synchronized. Constraint (10) takes care of the precedence relations and finally constraint (11) is used to calculate the total mission time.

This extended master problem is more complicated than the basic master problem, hence it might not be as easy to solve, but this model is still much smaller than the full problem from Section 4.2. We have not implemented this yet, so no results are available, but for the moderate problem sizes considered here, this new model should not pose any real problem for an advanced MIP solver.

8.5 Resolving Conflicts

The Extended Master Problem can be modified to handle flight path conflicts. If a solution does not pass the post processing check, due to an intersection or collision between two paths, it is possible to add constraints that resolve this conflict by forbidding the current solution. Consider the example in Figure 28, where the two flight paths intersect at some point X.

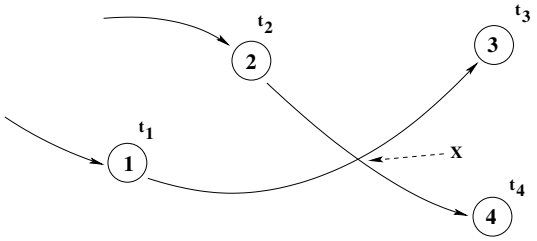


Figure 28: Two aircraft with flight paths that intersect. In the current solution both aircraft reach the intersection point X at the same time.

Based on a solution $(\bar{x}, \bar{t})$, the post processing check creates a detailed flight path for each aircraft, where each arc is broken down into smaller legs with check times, and can specify the estimated time of arrival to the intersection point X. If the estimated arrival times for the two aircraft to the intersection point are separated far enough in time, more than some user specified number T_{sep}, there is no real problem. On the other hand, if the difference in arrival times is small, the proposed solution is not OK due to the risk of collision.

Assume now that we do have a conflict. If one of the intersecting flight paths were changed in such a way that the actual time of intersection becomes acceptable, i.e. separated more than T_{sep}, the problem is solved. The question is which path that should be changed, or if it is optimal to change at least one of the current flight paths for some other solution.

It is possible to model and resolve this conflict by augmenting the model with a binary variable q, that specifies which flight path that should be altered, and two additional constraints

$$\begin{aligned} t_1 + T_{13}x_{13} + T_{sep} \cdot (x_{13} + x_{24} + q - 2) &\leq t_3 + (1 - x_{13}) \cdot T_{max} \\ t_2 + T_{24}x_{24} + T_{sep} \cdot (x_{13} + x_{24} - q - 1) &\leq t_4 + (1 - x_{24}) \cdot T_{max} \end{aligned}$$

which forces one of the paths to alter its traveling time *if* both paths are in use, that is $x_{13} = x_{24} = 1$. If $q = 1$ then path $(1, 3)$ is altered, and if $q = 0$ path $(2, 4)$ is altered.

Notice that if either $x_{13} = 0$ or $x_{24} = 0$, or both, the new constraints become redundant. This is important since the model should be able to find other subtour combinations if that is optimal subject to the new constraints. The solution to the new problem, including the above constraints, either consist of the same subtours as in the original solution z^*, but with altered times so that the conflict is avoided, or have changed to a different set of subtours.

To generalize this procedure, let $\mathcal{C}$ denote the set of conflicting arc pairs detected in the post processing check. For each such pair of conflicting arcs $[(i, j), (k, l)] \in \mathcal{C}$, we define the binary variable q as

$$q_{kl}^{ij} = \begin{cases} 1 & \text{if route } (i, j) \text{ should be altered} \\ 0 & \text{if route } (k, l) \text{ should be altered} \end{cases}$$

and add constraints

$$\begin{aligned} t_i + T_{ij}x_{ij} + T_{sep} \cdot (x_{ij} + x_{kl} + q_{kl}^{ij} - 2) &\leq t_j + (1 - x_{ij}) \cdot T_{max} \\ t_k + T_{kl}x_{kl} + T_{sep} \cdot (x_{ij} + x_{kl} - q_{kl}^{ij} - 1) &\leq t_l + (1 - x_{kl}) \cdot T_{max} \end{aligned}$$

to the model, a total of $|\mathcal{C}|$ binary variables and $2|\mathcal{C}|$ constraints, and solve the Extended Master Problem again. If the new solution passes the post processing check, we are done. Otherwise one keep adding constraints for each new conflict until there are none.

Notice that if T_{sep} is chosen very large, as a direct consequence, this specific combination of paths is forbidden and at least one of the routes will change.

8.6 Future Work

There are many possible extensions to the column based heuristic outlined here. At the moment, we use $n_k = 2$ aircraft, that is one aircraft pair to take care of the targets in each subset S_k. Conceptually, it is straightforward to generalize this and generate additional columns where $n_k = m$ aircraft are utilized to deal with the targets in each subset. Exactly how to solve each subproblem in an efficient way, using m aircraft instead of two, is presently not obvious.

The extended master problem presented in Section 8.4, and the additional constraints for resolving conflicts discussed in the previous section, are still to be implemented and tested. It is impossible to say whether this approach of resolving conflicts, by adding constraints, converges quickly or just keep generating new conflicts.

9 Benchmark

In order to test and compare the proposed heuristic methods, we define a set of problems to be part of a benchmark. In these tests we like to cover some different characteristics, like the number of targets, the size of the aircraft fleet, vary the line of entrance and exit, and also solve problems for different precedence relations.

9.1 Scenarios and Cases

We consider three different scenarios, involving five targets each, where the relative positions of the targets vary. The scenarios are referred to as 100, 200 and 300, and each scenario comes with three different versions of entry and exit lines, referred to as case 1, case 2 and case 3. The first scenario in combination with the third case of entry/exit lines are thus referred to as Scenario 103. The closest distance between targets range from 10 kilometers up to 25 kilometers. One unit in the pictures corresponds to 1 km.

Scenario 1

The first scenario is illustrated in Figure 29. The targets are positioned without any clear structure, but not too tight together. The first target is surrounded by three protected object, indicated here by green squares. The second target is protected by four SAM sites, seen as red circles, which affects the expected target effect from certain directions. There are also two protected objects in its vicinity. In a similar manner, the other targets are also heavily guarded in some directions, together with a few protected objects.

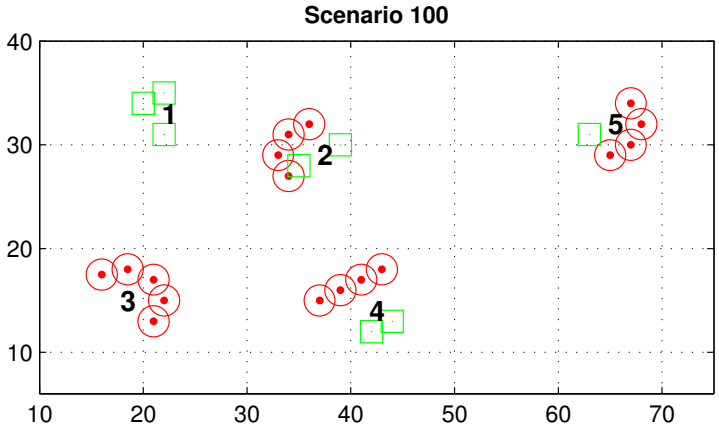


Figure 29: Scenario 1.

The discretization of this scenario is found in Figure 30, with a total of 66 attack nodes (black dots), 44 illumination nodes (black plus) and a total of 9550 arcs (not shown). These numbers are the same for all scenarios. There are $G = 6$ sectors for each target, illustrated by the dashed lines and circes in blue. The results presented here are found by the constructive heuristic, but a comparison between the different heuristics are found later in this section.

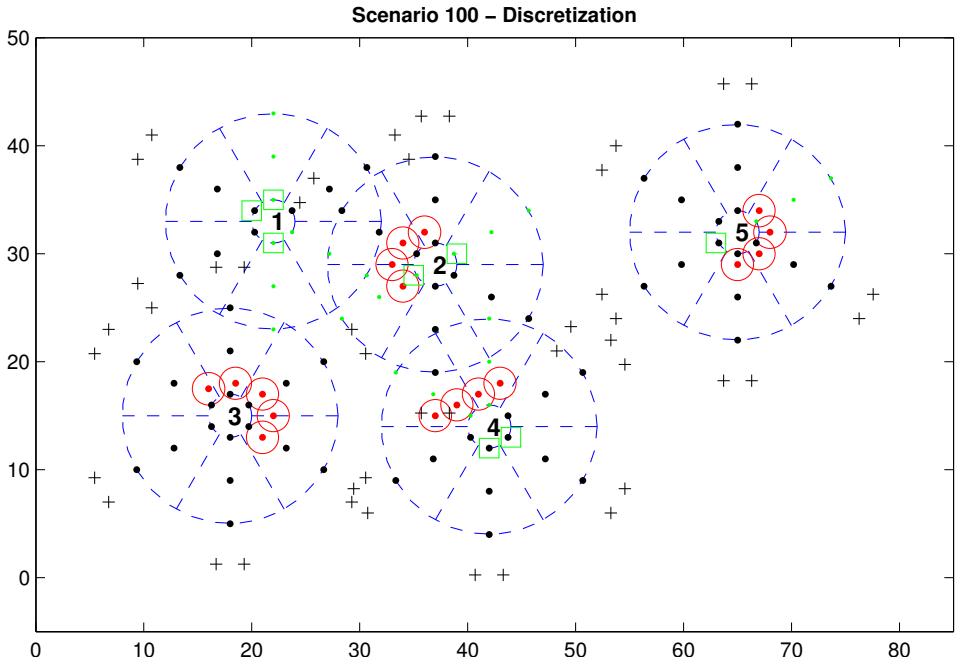


Figure 30: Discretization of Scenario 1, which consist of 66 attack nodes (black dots), 44 illumination nodes (black plus) and a total of 9550 arcs (not shown).

Scenario 2

The second scenario is illustrated in Figure 31, where the targets are positioned in a straight line. Its discretization is not shown here, but consist of exactly the same number of nodes and arcs as for the first scenario. This is due to identical vicinity around each target, it is only the targets relative positions that are changed.

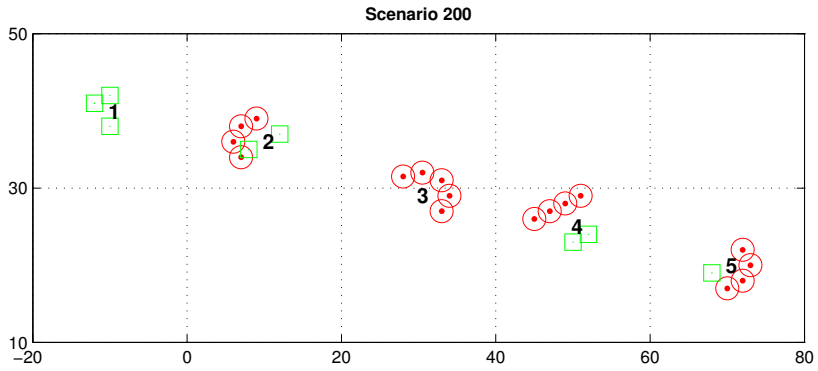


Figure 31: Scenario 2.

Scenario 3

The third scenario is illustrated in Figure 32. Here the targets are positioned in a v-shaped formation. Its discretization is not shown.

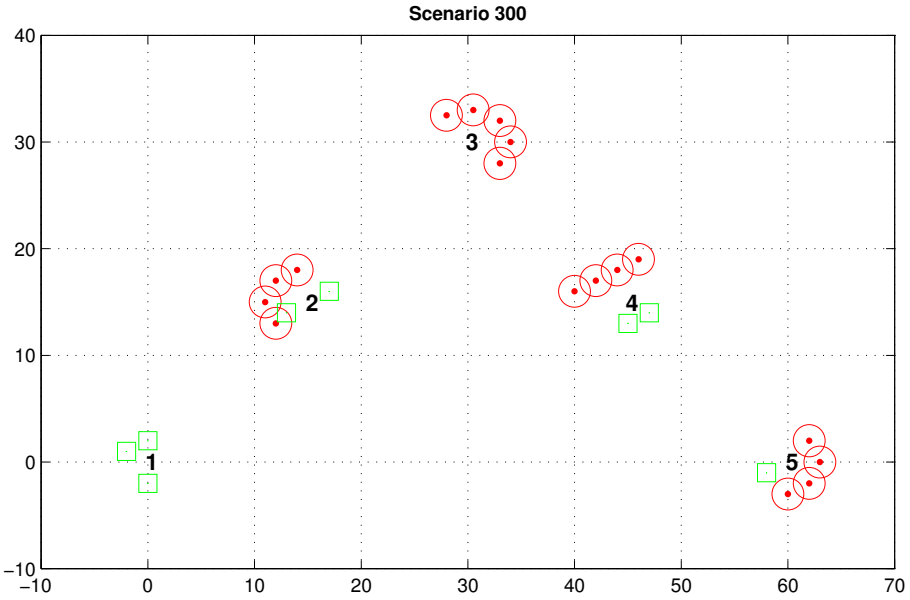


Figure 32: Scenario 3.

9.2 Problem Instances

For illustration purposes, this section provide solutions for numerous scenarios with varying assumptions on the aircraft fleet, and the problem instances are solved by the Constructive Heuristic method. We consider the following cases:

Geometry

In the geometry section, we demonstrate that the model and heuristics are able to provide good solutions for various scenarios, where the relative positions between targets are changed, but also with respect to the entry and exit lines. We solve each of the three scenarios for different positions of the entry line and exit line and present results for two aircraft.

Specified Roles

We present results where a certain number of aircraft are specified to be attackers and the other aircraft are specified to be illuminators. These assumptions cannot be solved using the Column Enumeration approach at the moment, but results are reported for the Constructive Heuristic and the mathematical model solved by AMPL/CPLEX.

Fleet Size

The model and heuristics should be able to utilize multiple aircraft in efficient ways, hence the first scenario is solved for an increasing aircraft fleet.

Precedence Constraints

The model and heuristics are required to handle precedence constraints, and four sets of precedence constraints are prescribed to the first scenario. Each instance is solved for two and four aircraft to investigate the cooperative gains of multiple aircraft.

Run times for the Constructive Heuristic varies between one second up to at most ten seconds. Later, in Section 9.7, benchmark results are presented where the Constructive Heuristic and the Column Enumeration approach are compared, together with results for AMPL/CPLEX.

Interpretation of Results

The entry and exit lines are represented by black solid lines and recognized by the text IN and OUT respectively. Target locations are indicated by the BLACK numbers, which also gives a natural numbering of the targets. Red

circles indicate enemy defensive positions, and their radius specify the area covered. Protected objects are shown as green squares. The flight path for each aircraft is shown as a dashed line, and attack and illumination nodes along these paths are indicated by dots. In general, the black dashed routes correspond to attacks and the blue dashed routes to illuminations.

The title of each figure state the scenario number and the *target sequence*, which specifies the proposed attack order of the targets found by the Constructive Heuristic.

9.3 Geometry

Here follows 9 figures, presenting solutions to the different scenarios and positions of the entry and exit lines.

Scenario 100

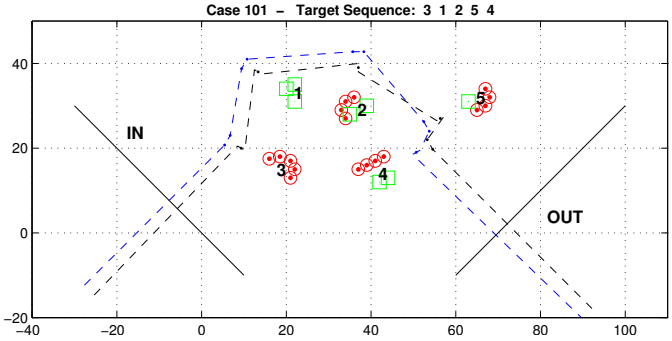


Figure 33: Result for Case 101 using 2 aircraft. Mission time $T_F = 689$ and expected effect $pKill = (1.0, 1.0, 1.0, 0.8, 1.0)$ against the targets.

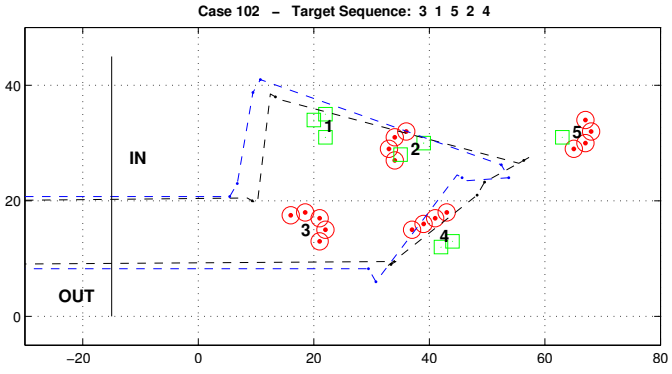


Figure 34: Result for Case 102 using 2 aircraft. Mission time $T_F = 948$ and expected effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ against the targets.

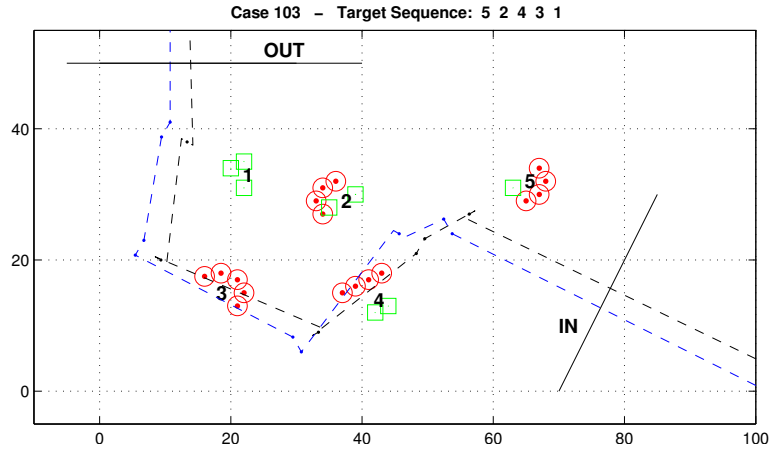


Figure 35: Result for Case 103 using 2 aircraft. Mission time $T_F = 676$ and expected effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ against the targets.

The different placements of the entry and exit lines are captured by the model, and as seen in the solutions the target sequence is adjusted in order to achieve time efficient flight paths. High expected effect is obtained against all targets.

Scenario 200

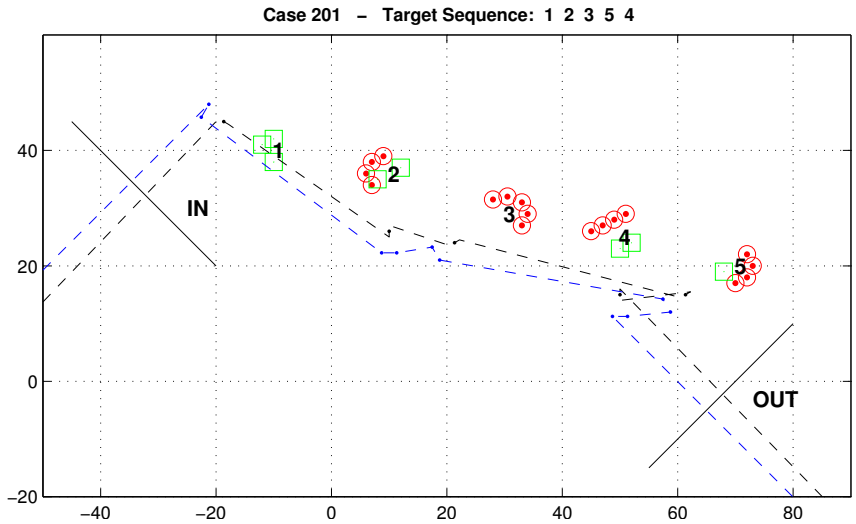


Figure 36: Result for Case 201 using 2 aircraft. Mission time $T_F = 844$ and expected effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ against the targets.

Just like in the case of Scenario 100, the different placements of the entry and exit lines are captured by the model. The target sequence is adjusted in order to achieve time efficient flight paths. High expected effect is obtained against all targets.

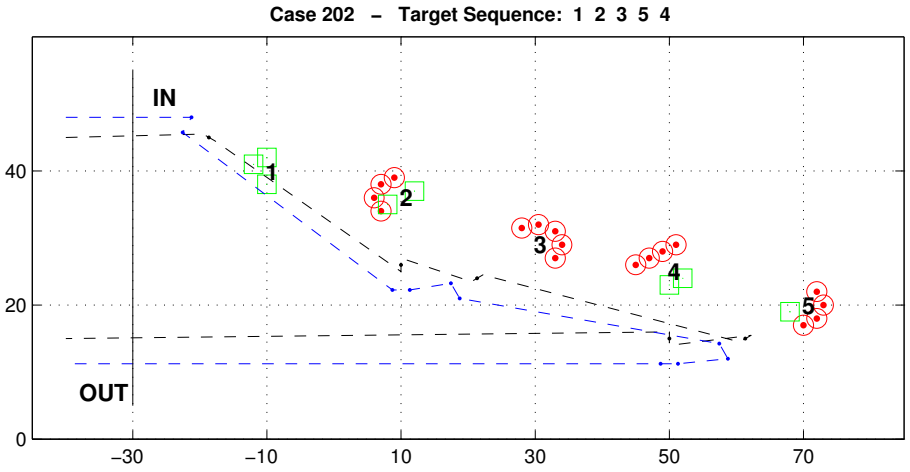


Figure 37: Result for Case 202 using 2 aircraft. Mission time $T_F = 1074$ and expected effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ against the targets.

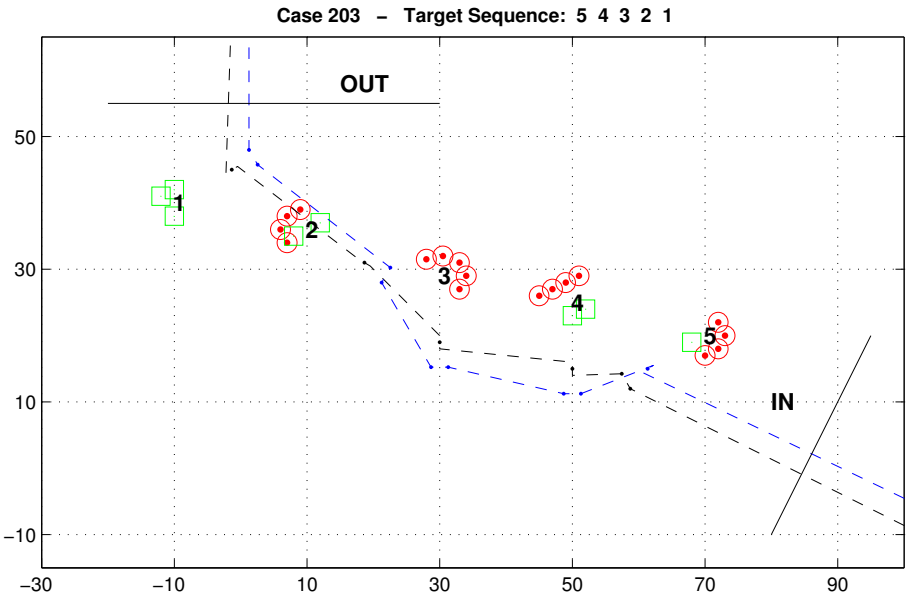


Figure 38: Result for Case 203 using 2 aircraft. Mission time $T_F = 612$ and expected effect $pKill = (0.92, 1.0, 1.0, 1.0, 1.0)$ against the targets.

Scenario 300

The results are once again similar to those for Scenario 100. The target sequence is adjusted in order to achieve time efficient flight paths, and high expected effect is obtained against all targets.

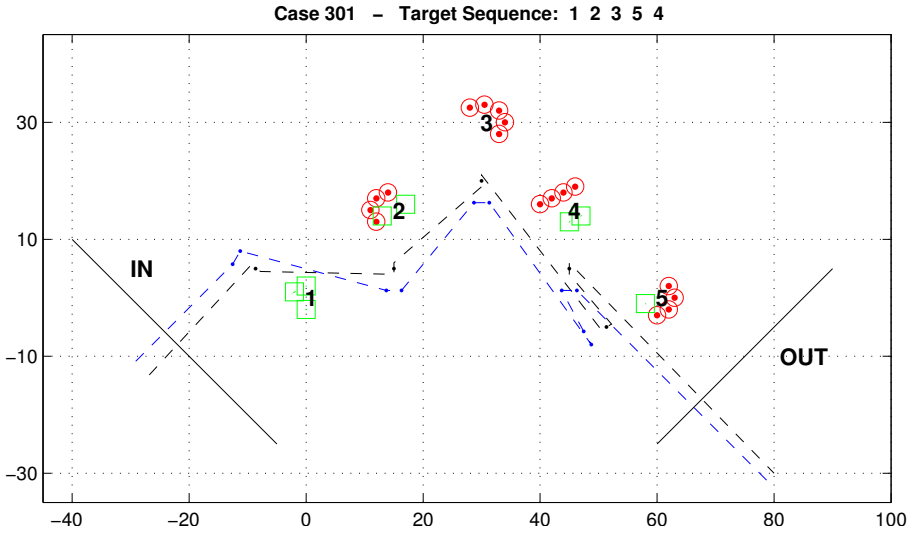


Figure 39: Result for Case 301 using 2 aircraft. Mission time $T_F = 752$ and expected effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ against the targets.

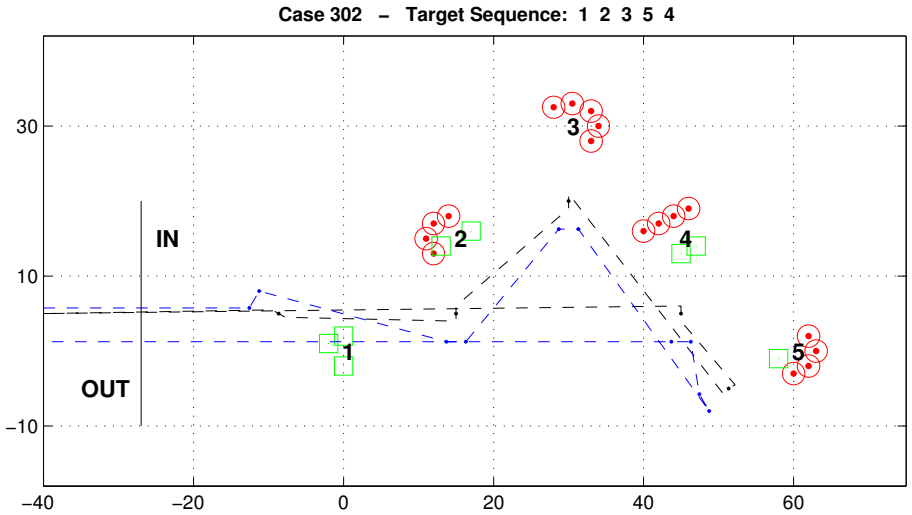


Figure 40: Result for Case 302 using 2 aircraft. Mission time $T_F = 988$ and expected effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ against the targets.

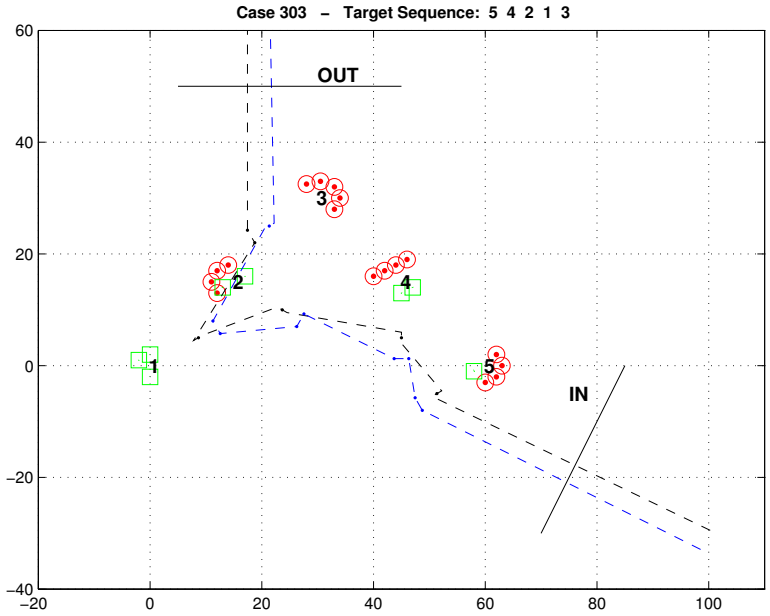


Figure 41: Result for Case 303 using 2 aircraft. Mission time $T_F = 693$ and expected effect $pKill = (0.92, 1.0, 1.0, 1.0, 1.0)$ against the targets.

9.4 Specified Roles

For these tests, each aircraft is given a specified role throughout the mission. In the title of each figure, the number of attackers plus the number of illuminators is stated.

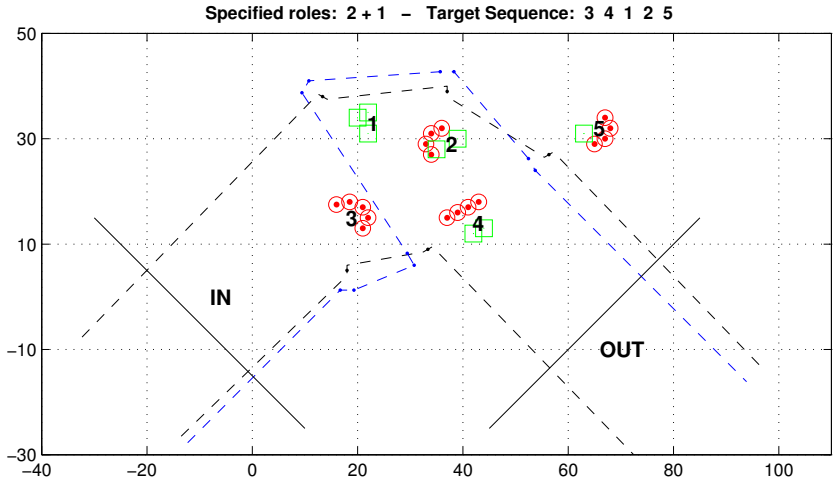


Figure 42: Result for Case 101 with 2 attackers and 1 illuminator. Expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ and mission time $T_F = 819$.

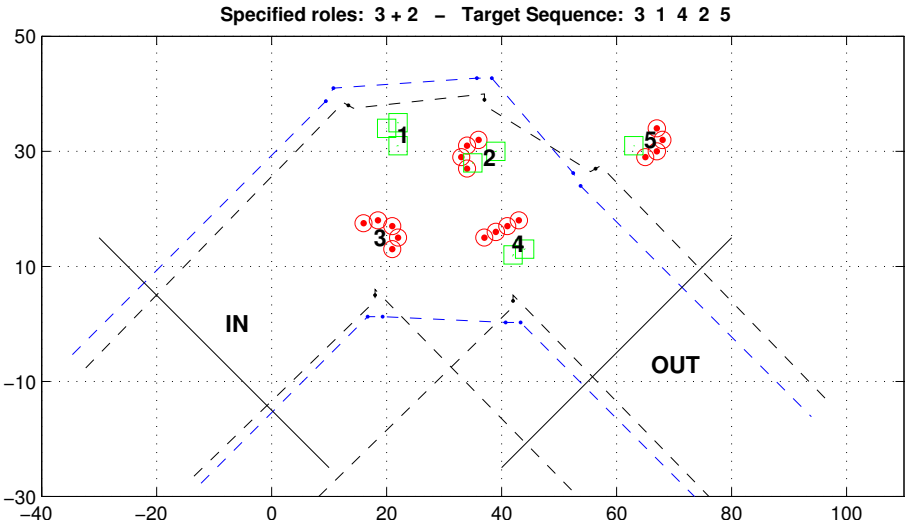


Figure 43: Result for Case 101 with 3 attackers and 2 illuminators. Expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ and mission time $T_F = 594$.

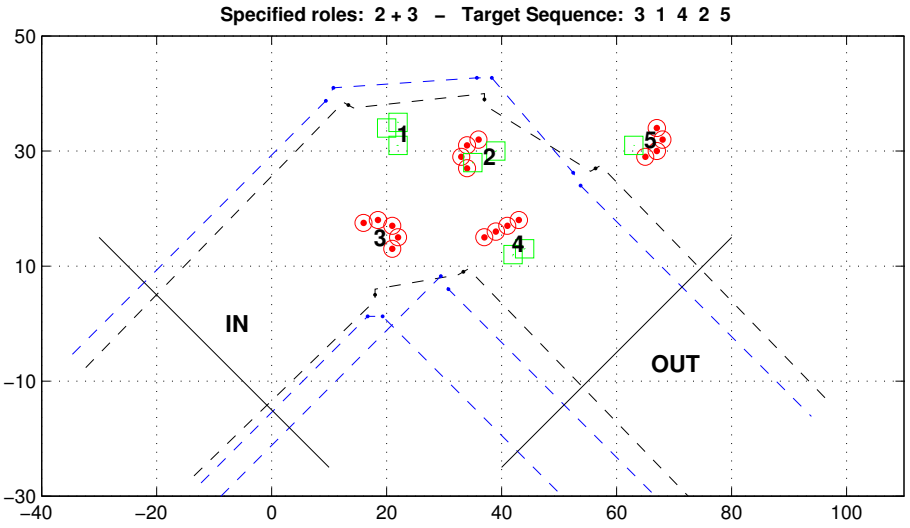


Figure 44: Result for Case 101 with 2 attackers and 3 illuminators. Expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ and mission time $T_F = 594$.

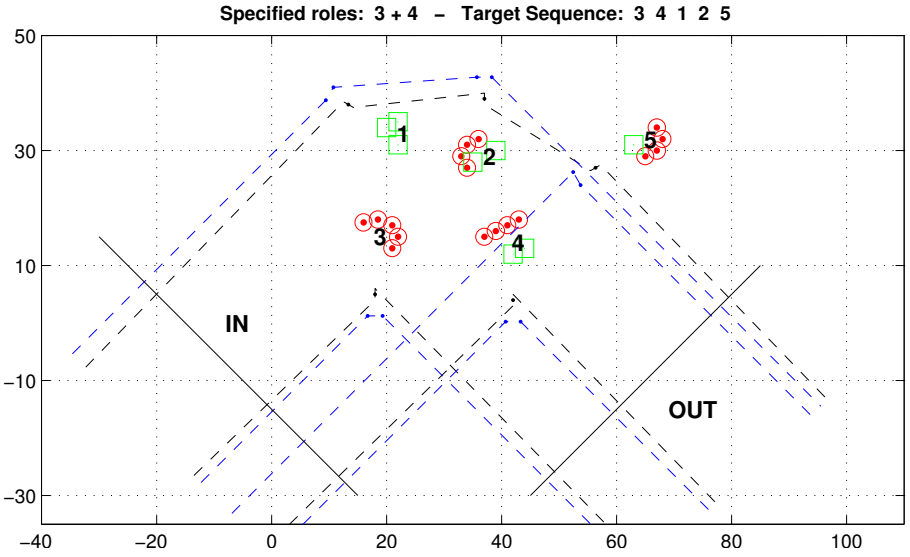


Figure 45: Result for Case 101 with 3 attackers and 4 illuminators. Expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ and mission time $T_F = 586$.

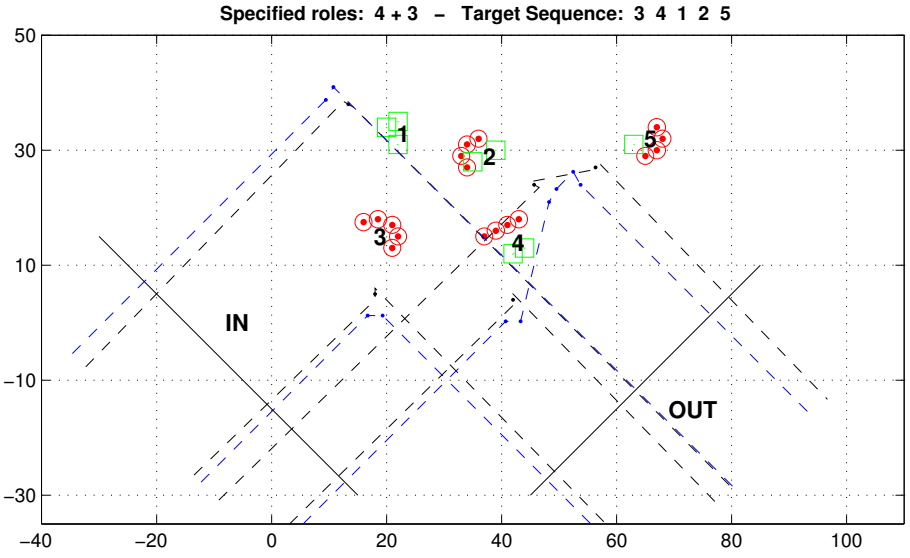


Figure 46: Result for Case 101 with 4 attackers and 3 illuminators. Expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ and mission time $T_F = 498$.

It is beautiful to see how an increasing aircraft fleet better and better partitions the targets and decreases the total mission time.

9.5 Fleet size

Here we solve Case 101 once again, but now the number of aircraft used in the mission is varied. As seen in the solutions, an increased number of aircraft results in smoother flight paths, as the targets are partitioned between the resource pairs. The total mission time naturally decreases as the fleet size increases.

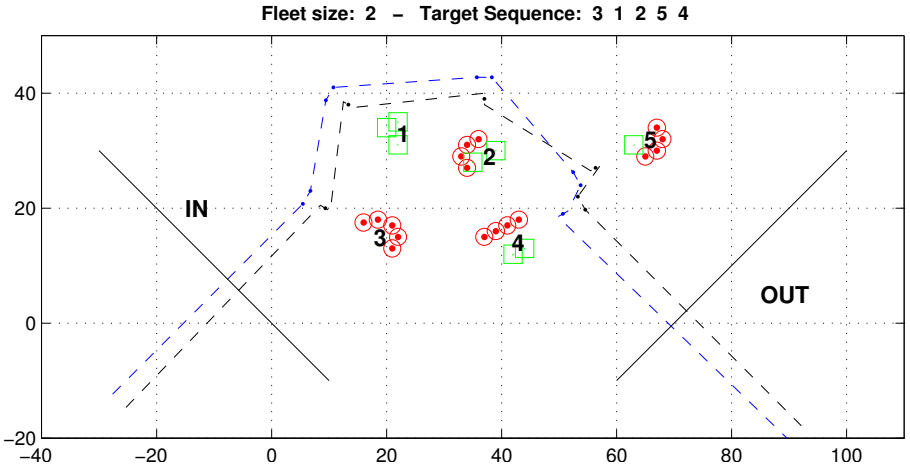


Figure 47: Result for Case 101 using 2 aircraft. Mission time $T_F = 689$ and expected target effect $pKill = (1.0, 1.0, 1.0, 0.8, 1.0)$.

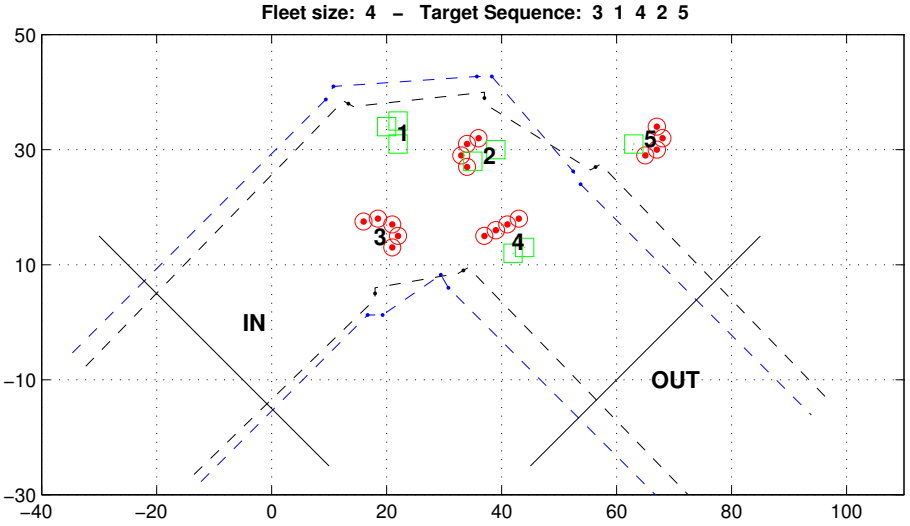


Figure 48: Result for Case 101 using 4 aircraft. Mission time $T_F = 594$ and expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$.

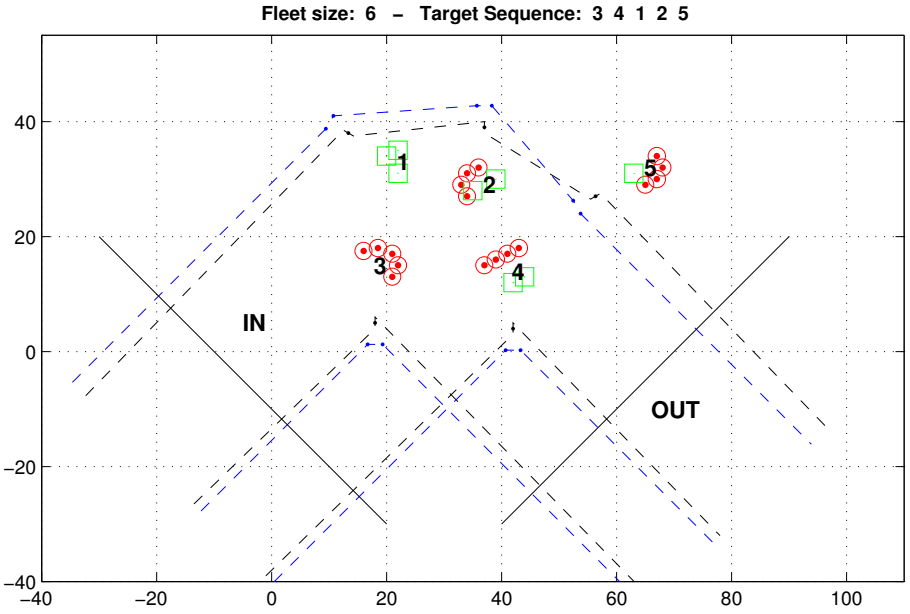


Figure 49: Result for Case 101 using 6 aircraft. Mission time $T_F = 594$ and expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$.

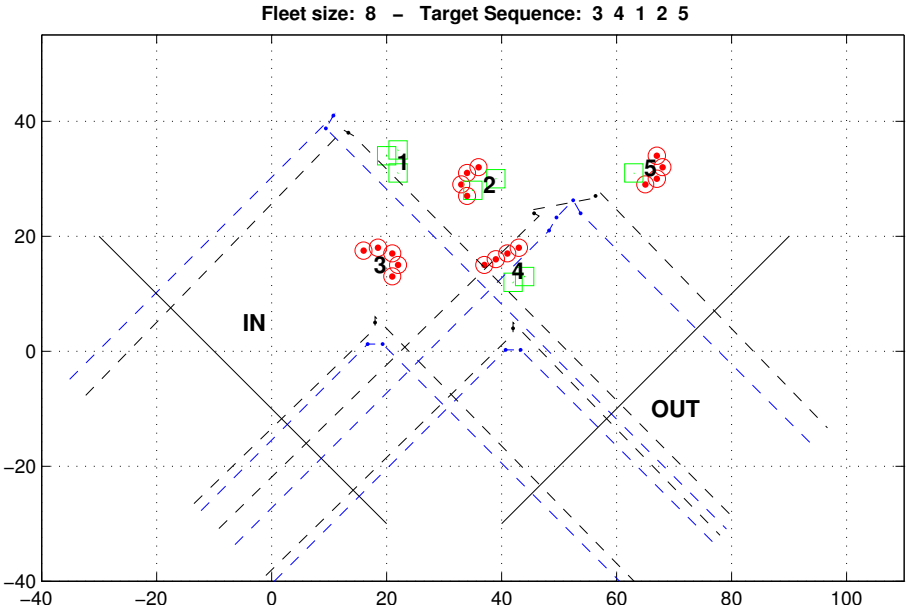


Figure 50: Result for Case 101 using 8 aircraft. Mission time $T_F = 459$ and expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$.

9.6 Precedence Constraints

For these scenarios, we specify an attack sequence, partially or totally ordered, to show that the heuristic is able to handle this. Each case is solved using two aircraft and four aircraft respectively, to highlight the efficient use of additional aircraft.

Enforced Sequence: 1–2–3–4–5

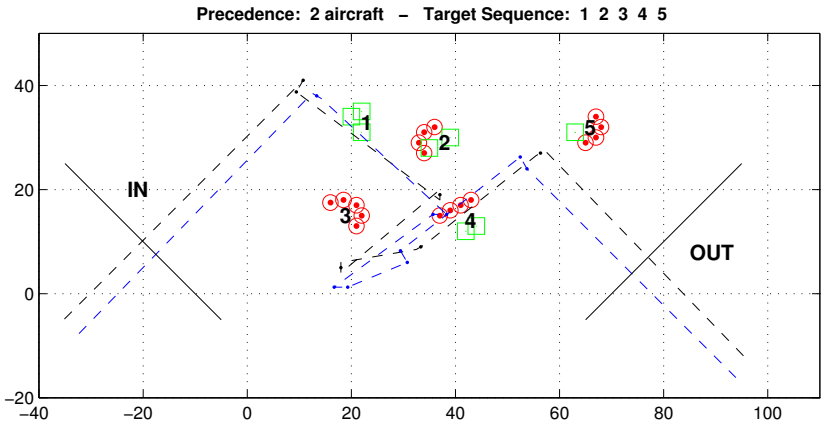


Figure 51: Result for 2 aircraft with precedence 1–2–3–4–5 on targets. Expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ and mission time $T_F = 931$.

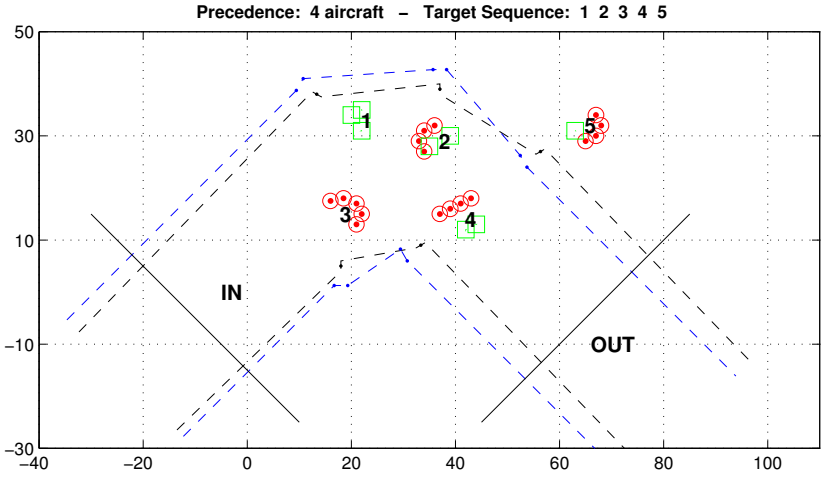


Figure 52: Result for 4 aircraft with precedence 1–2–3–4–5 on targets. Expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ and mission time $T_F = 594$.

The enforced sequence gives rise to long flight paths in the case of two aircraft. Adding an extra pair of aircraft helps a lot as the targets are partitioned in such a way that each pair can travel in a natural path from the entry line to the exit line. This decreases the the total mission time drastically. Similar behaviour is seen for all cases.

Enforced Sequence: 1–{2,3}–{4,5}

Here target 1 must be attacked before targets 2 and 3, whom in their turn must be attacked before targets 4 and 5. There is no restriction on the order of attack between targets 2 and 3, and similar between targets 4 and 5.

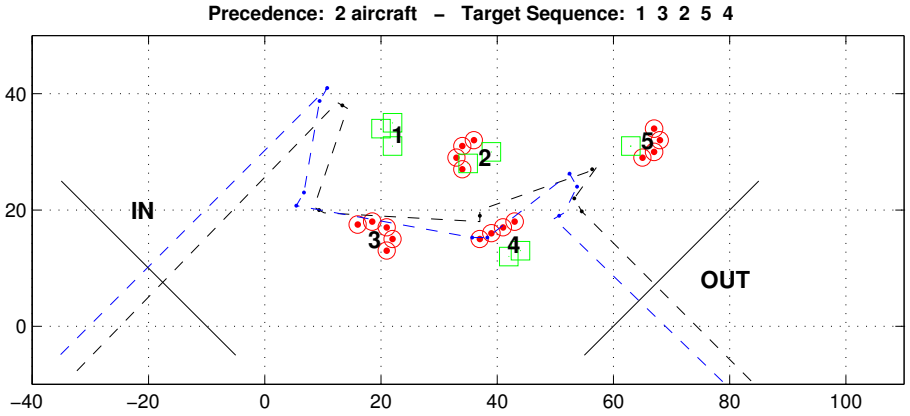


Figure 53: Result for 2 aircraft with precedence 1–{2,3}–{4,5} on targets. Expected target effect $pKill = (1.0, 1.0, 1.0, 0.8, 1.0)$ and mission time $T_F = 757$.

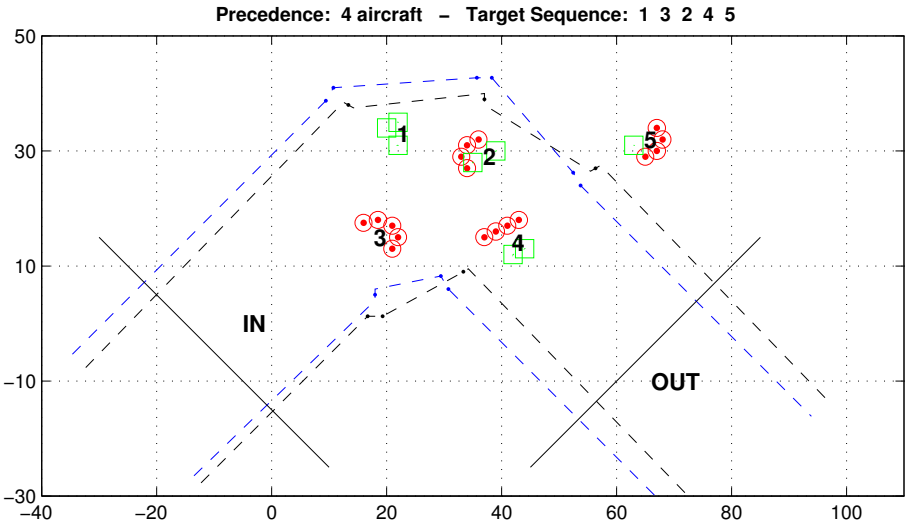


Figure 54: Result for 4 aircraft with precedence 1–{2,3}–{4,5} on targets. Expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ and mission time $T_F = 594$.

Enforced Sequence: 1–3–4–2–5

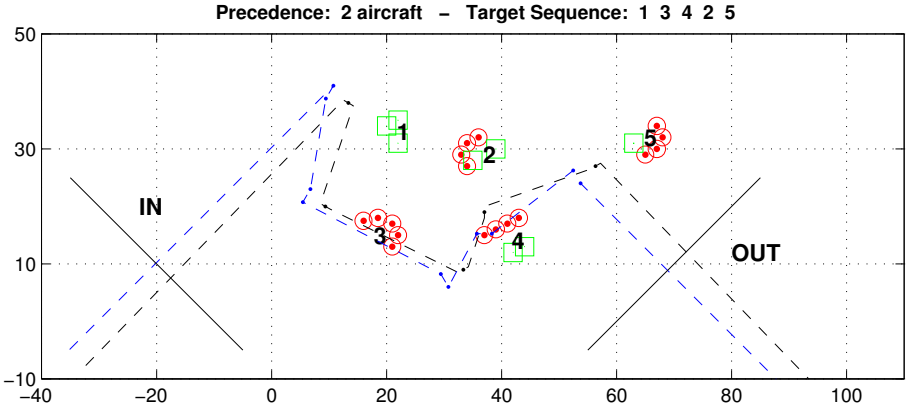


Figure 55: Result for 2 aircraft with precedence 1–3–4–2–5 on targets. Expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ and mission time $T_F = 779$.

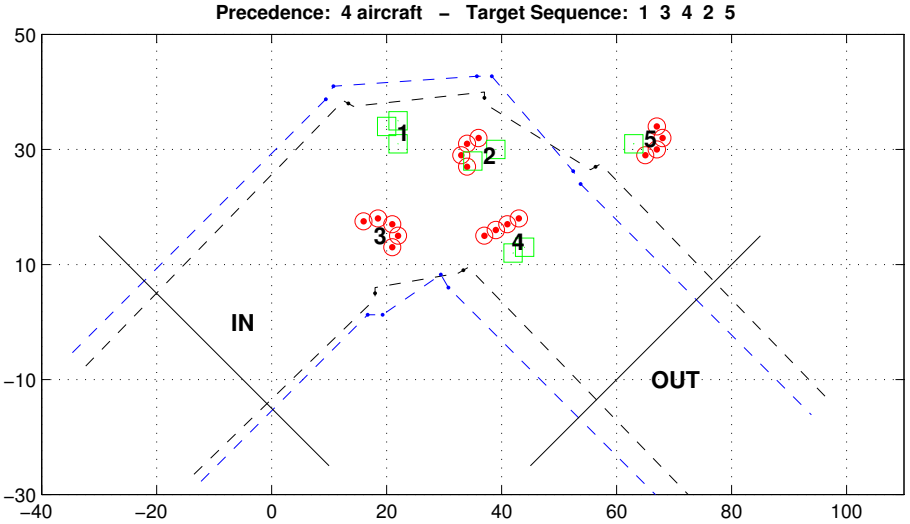


Figure 56: Result for 4 aircraft with precedence 1–3–4–2–5 on targets. Expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ and mission time $T_F = 594$.

Once again, the enforced sequence gives rise to long flight paths in the case of two aircraft. With an extra pair of aircraft, the targets are partitioned in such a way that allows each pair to travel in a natural path from the entry line to the exit line. This decreases the the total mission time drastically.

Enforced Sequence: {3,4}–{1,2,5}

Here targets 3 and 4 must be attacked before targets 1, 2 and 5. There is no restriction on the order of attack between targets 3 and 4, and similar between targets 1, 2 and 5.

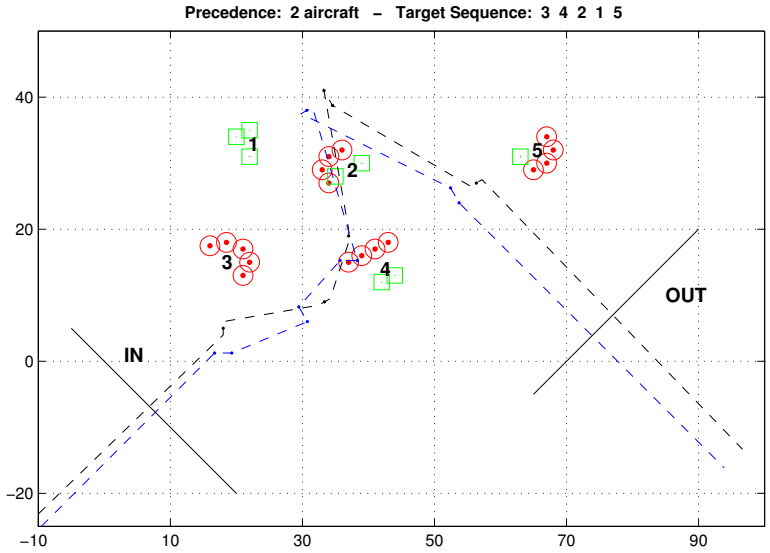


Figure 57: Result for 2 aircraft with precedence {3,4}–{1,2,5} on targets. Expected target effect $pKill = (0.92, 1.0, 1.0, 1.0, 1.0)$ and mission time $T_F = 702$.

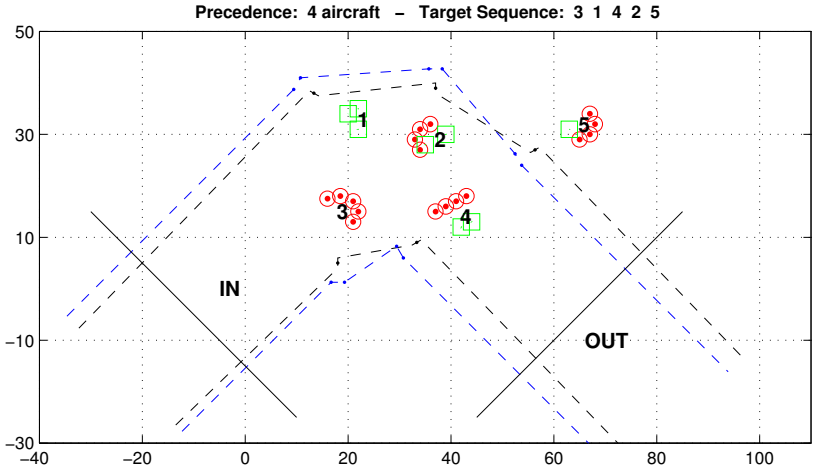


Figure 58: Result for 4 aircraft with precedence {3,4}–{1,2,5} on targets. Expected target effect $pKill = (1.0, 1.0, 1.0, 1.0, 1.0)$ and mission time $T_F = 611$.

The enforced partially ordered sequence is handled in a good way by only two aircraft, but the extra pair still improves the total mission time and expected effect.

9.7 Method Comparison

In order to compare the two heuristics, we have used the Column Enumeration approach to solve the same scenarios as in the previous section. We have also used the option where columns are generated heuristically, to see whether the extra effort of evaluating all combinations is motivated or not. The results are presented in Table 4, where the objective value is given, together with reference values found using AMPL/CPLEX for a limited time.

The results for the Constructive Heuristic is presented in column **Constr**, and the solutions are the ones presented in the previous section. The Column Enumeration approach is presented in columns **Greedy** and **Enum**, where the former refers to the version of the method where columns are found heuristically, and the latter where all combinations are tried. For these tests, in order to compare the results with the Constructive Heuristic, we use no capacity restriction on the aircraft fleet. Also, we solve the subproblems for the set of all five targets, since most instances only involves one aircraft pair.

Constructive Heuristic

The Constructive Heuristic (CH) generate a feasible solution within 1–5 seconds for any problem instance. As seen in Table 4, they are never the best found solutions, but within its timeframe the results are outstanding. For problem instance 205, it finds a solution that is actually better than what CPLEX finds within 10 minutes. By doing some fine tuning of this method, experimenting on how to weight the arc costs, the Constructive Heuristic will be an important routine in developing more advanced heuristics.

Greedy Column Generation

Solution times for generating the greedy columns are 12–26 seconds, where problems with precedence constraints run faster due to smaller search space. The time needed for solving the master problem is neglectable, normally a few hundredths of a second. This is interesting for problem instances where the fleet size vary, since the columns only need to be generated once.

The Greedy Column Generation is based on the Constructive Heuristic, and thus their results are identical for problem instances with two aircraft. Hence the Greedy Column Generation approach is only suitable for instances where more than one aircraft pair is available. This is seen in Table 4 for the fleet size instances, where better solutions are found by the master problem when combining columns, compared to the Constructive Heuristic.

When solving problem instances with precedence constraints, better solutions are still found, but the improvement is less impressive. For instance **Prec 1.4**, the result is actually worse than for the Constructive Heuristic. This is due to the precedence constraints, as their effect is not seen before

Table 4: Model comparison. The Constructive Heuristic (Constr), the heuristically found columns (Greedy) and the Column Enumeration (Enum). Also the best found AMPL/CPLEX solution after 1 minute and 10 minutes.

	HEURISTICS			AMPL/CLPEX	
	Constr.	Greedy	Enum.	1 min	10 min
Geom.					
101	129.26	129.26	122.33	133.19	124.28
102	145.57	145.57	135.63	144.80	137.92
103	130.26	130.26	119.98	127.43	123.67
201	145.17	145.17	129.79	141.85	129.99
202	171.19	171.19	164.69	171.98	165.11
203	125.55	125.55	125.55	138.13	127.59
301	142.69	142.69	128.57	136.71	129.00
302	158.10	158.10	149.34	159.10	152.60
303	134.00	134.00	132.62	145.05	133.99
S.R.					
2+1	141.74	-	-	121.63	121.63
3+2	120.75	-	-	115.01	114.41
2+3	120.81	-	-	117.98	110.27
3+4	121.95	-	-	125.85	110.19
4+3	113.63	-	-	114.49	109.89
Fleet					
2	129.26	129.26	122.33	133.19	124.28
4	120.07	109.00	107.75	112.95	108.93
6	121.36	105.30	104.85	114.23	110.48
8	110.64	105.02	104.55	119.31	112.70
Prec.					
1.2	153.65	153.65	136.36	138.58	138.58
1.4	120.62	133.98	111.68	118.63	111.24
2.2	136.42	136.42	124.28	124.73	124.21
2.4	120.40	117.48	108.73	118.82	108.73
3.2	137.68	137.68	128.77	131.19	131.19
3.4	120.25	120.00	115.29	122.58	114.13
4.2	130.06	130.06	129.45	133.87	129.45
4.4	121.73	109.22	109.22	126.78	110.61

validating the solution, that is, solving the project network. The combination of two or more columns suggested by the master problem are surely optimal if there were no precedences, but in order to create a feasible solution the total mission time might need to be prolonged. This behaviour was predicted in Section 8.3, and is illustrated in Figure 27 on page 49.

Section 8.3 also describes how to circumvent this behaviour, by iteratively adding constraints that forbid certain combinations of columns, and thus find alternative column combinations. For problem instance 1.4, the new objective became 119.39, which is better than for the Constructive Heuristic.

By extending the Greedy Column Generation to incorporate columns for each ordering of targets, and iteratively add constraints in order to find alternative column combinations, this approach will become more stable for problem instances with precedence constraints.

Complete Column Enumeration

For the complete enumeration of columns, solution times vary a lot depending on the precedence constraints. For problem instances with no precedence at all, it takes roughly 3 hours to generate all columns, but for a problem with fixed attack sequence the columns are found within 6 minutes. It is important to remember that these run times are totally dependent on the available computer capacity and implementation, and not that interesting since the process is totally parallelizable.

For problem instances with no precedence, and with increasing fleet size, the results are equivalent to the Greedy approach. This indicates that for subsets with only two or three targets, the Constructive Heuristic solutions are very good, but that these solutions become weaker for larger subsets. When precedence constraints are active, big improvements are found even for instances with two aircraft. This is very beneficial since the time needed for generating the columns decrease for exactly these problem instances.

To conclude, even though the Column enumeration requires long run times, its solutions are excellent and most of the time even better than what CPLEX finds after 10 minutes. By guiding the solution process of the heuristic, by excluding the most unlikely target orderings in a smart way, a substantial speedup is possible.

Heuristics versus AMPL/CPLEX

Based on the empirical tests in Section 5.5, the results for CPLEX are found using the second model, and we note the objective value found by CPLEX after 1 minute and after 10 minutes. Although far from closing the duality gap, it is able to provide feasible solutions within seconds, and it finds good solutions within 1 minute even for these bigger problem instances.

Even though CPLEX delivers the best results, sometimes within a minute, the heuristic methods are still motivated. The solutions found by the Constructive Heuristic, within a few seconds, are superior to those of CPLEX within that timeframe. For the application of military decision support, where it is essential to find good solutions quickly rather than optimal solutions after long time, this approach is favorable. It should also be stated

that the CH will always deliver a solution within seconds, whereas there is no guarantee for how quickly CPLEX will find a feasible solution.

The Column enumeration approach is well suited for problem instances with precedence constraints, since this limits the number of subproblems to be solved significantly. If only one resource pair is available, these solutions are probably near-optimal and found in reasonable time. For multiple aircraft pairs, the validation of the proposed column combination might render time inefficient solutions, but this seems to be a bigger issue for the Greedy Column Generation approach.

As for now, all heuristics are implemented in Matlab, and a professional implementation in C++ would surely increase their efficiency. This is especially true for the Column Enumeration approach which includes many for-loops, a known weakness for Matlab, and also due to its parallelizability.

9.8 Result Analysis

Flight paths

In Section 3.5 we described how arc costs are calculated. Although an arc is presented as a straight line in the network representation, we remind you that it actually consist of a complex and detailed flight path. For example, in the extracted part of a solution found in Figure 59, the flight path between the attack nodes is represented by a straight line, seemingly passing through the defended airspace of a SAM. The actual flight path avoids the SAM and takes advantage of the environment in order to stay hidden.

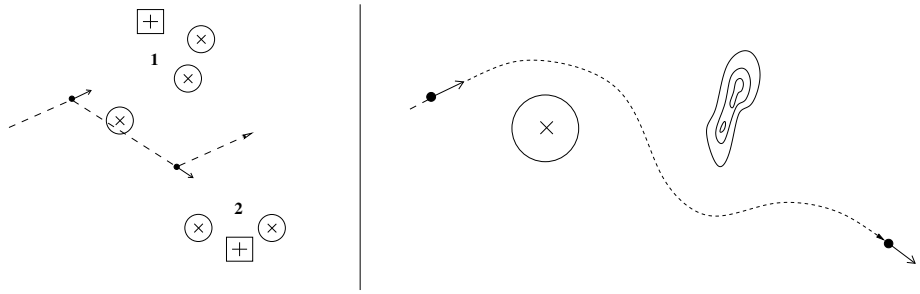


Figure 59: The flight path of an aircraft. In solutions, each arc represents a complex flight path between its nodes.

9.8.1 Deconfliction

Our general mission planning problem has been solved, and by some inductive reasoning, we can state that our models and methods hold and provide "good" solutions as seen in the solution figures. A "good" solution is indicated by fulfillment of the objectives, i.e. a high hit probability in

combination with time efficient routing. To find high hit probabilities one has to investigate the parameterized target area and search for good candidates. In our model set up, it is straightforward to identify preferred attack points and directions against each target.

These attack points are connected to paths, one path for each resource. As mentioned earlier, hit probability is easy to verify but how can the paths be examined? One can easily see the geometric aspect of the solution. Is the route smooth, does it look like a shortest path between the entrance and exit line, then it is a good solution. However, since there can be a conflict between finding the shortest path from entrance to exit line and still catch the high probability attack points, the solution might look inefficient in some cases, specially if we invoke strong precedence constraints. Some of these aspects can be visually examined, some can not. Ways of stating objective fulfillment is of course by examine the objective value and the degree of optimality. As showed in this section, the Constructive Heuristic is effective and brings near optimal or optimal solutions, so we can be confident that our balance of expected target effect and time efficient routing provide "good" solutions.

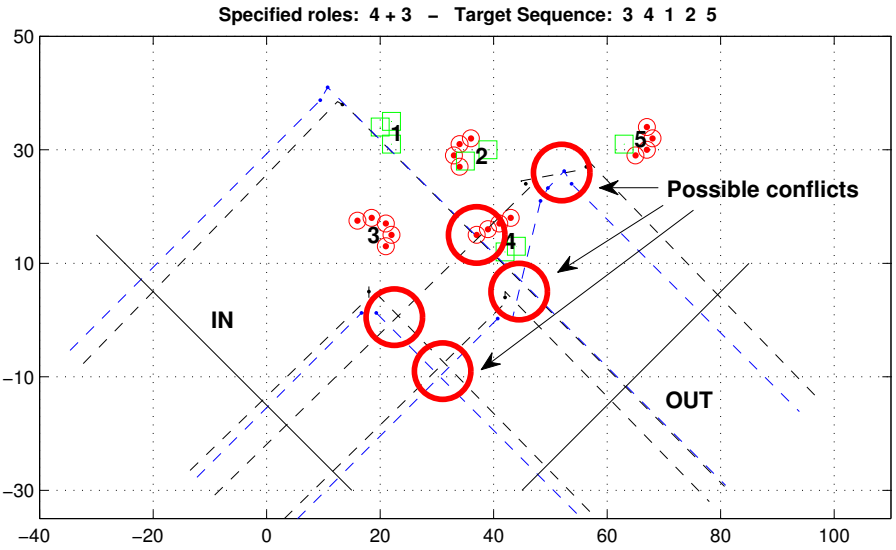


Figure 60: Deconfliction. A mission plan with possible conflicts.

But there is still one aspect of a mission plan that is vital and not yet covered by our optimization approach – is the plan conflict free? Our solution can contain conflicts in the sense that two or more paths coincide in time and space, hence invoke a probability of collision. A first suspicion of conflict is shown in Figure 60 where five path crossings occur. Further, a timing analysis calculates the relative distances within the mission time span, that is from the line of entrance to the exit line. If the distance between each pair of aircraft is less than some given security distance, a confliction is at hand.

Now a deconfliction process is carried through based on two different strategies. First deconfliction can be done by the separation of paths into different altitudes. An altitude difference is invoked in each confliction zone until all conflicts are solved. This can be done iteratively. A second approach to resolve conflicts is to separate flight paths in time. However, precedence constraints can be a hindrance in that process. To overcome precedence difficulties, individual legs can be shifted in time by a change in speed or a change in the local path layout.

10 Conclusions

In this paper, we have presented and formulated a military aircraft mission planning problem, involving the routing of an aircraft fleet that shall perform attacks against a given set of ground targets. The target scene might also include hostile SAM sites and protected objects, like schools and hospitals, not to be hit during the attacks. Feasible attack positions are defined by its footprint, the area subject to collateral damage, as positions such that no protected buildings fall inside the footprint.

The feasible airspace is discretized and, as a result, attack positions and compatible illumination positions can be represented by nodes, while feasible flight paths between these nodes define arcs. This network construction yields a mathematical model that is recognized as a generalized vehicle routing problem with several side constraints. Each target needs to be visited by exactly two aircraft in a compatible manner, that is, the attack and illumination positions should match. Further, the attack and illumination actions should be synchronized in time. Finally, precedence constraints restrict the order of the attacks.

In Section 4 we have presented two mathematical MIP models which incorporate the size of the aircraft fleet, precedence relations and specified roles. Without any loss of generality, additional constraints are introduced in order to eliminate some symmetry, and also to enable the option of forcing the aircraft to work in pairs. The models are validated in Section 5, where a small test scenario is solved under different assumptions. The results are illustrative and motivates the automatization of military aircraft mission planning.

Although valid for large problem instances, the direct application of a general MIP solver to the mathematical models is only practical for smaller scenarios. Efficient heuristics have therefore been developed, with great result, and the details for the Constructive Heuristics are found in Section 7 and the Column enumeration approach is found in Section 8.

In Section 9, the Constructive Heuristic was used to provide solutions to multiple scenarios under a variety of different assumptions. The solutions are all intuitively appealing, with high expected effect of the attacks and no

obvious flaws, and generated within seconds. In order to compare the heuristic approaches, all problem instances were also solved using the Column Enumeration and the Greedy Column Generation. This comparison have been presented in Section 9.7, together with solutions from AMPL/CPLEX found using the second MIP model. The results are convincing, as the Constructive Heuristic finds acceptable solutions within seconds and the Column Enumeration approach finds the overall best solutions for most instances.

Future Work

As always, there are many things that can be improved and explored further. If we look back at Figure 8 on page 12, we have only treated the Optimization Block, and even so not investigated and implemented all of its sub-blocks. The Preprocessing block consists of many challenging optimization problems, such as the sequencing aspects which results in difficult linear ordering problems.

The Post processing is also an important part, since the deconfliction step is absolutely necessary in order to validate the solutions. Without this step, the automatization of the military aircraft mission planning process is not possible, and limits the use of our work.

The heuristic approaches are not able to handle all the constraints and limitations covered by the mathematical models. For example, both the Constructive Heuristic and the Column Enumeration assume that aircraft work in pairs. There is one exception, as the Constructive Heuristic can handle a specified number of attackers and illuminators, but this is limited to the case where each aircraft has a specified role.

Moreover, the Constructive Heuristic cannot handle armament constraints, although this extension have been discussed in Section 7.5 and its implementation should be straightforward. It would also be interesting to modify the heuristic to solve each problem for different arc costs, specified by weights for the attack efficiency and time consumption, as discussed in the same section.

For the Column Enumeration approach, in Section 8.4 we have outlined an Extended Master Problem in order to handle precedence constraints in a more robust way, and a framework for resolving flight path confliction in Section 8.5. We hope to implement them both in the near future.

Ideas for a Lagrangean relaxation of the problem have been discussed, and many different relaxations are possible. It is important though to get subproblems that are tractable in some sense, either with well-known structures that can be utilized, or as problems with available and efficient solvers.

References

[1] R. Baldacci, E. Bartolini, and G. Laporte: Some applications of the generalized vehicle routing problem. *Journal of Operational Research Society* **61**, 1072–1077 (2010).

[2] M. C. Bartholomew-Biggs, S. C. Parkhurst, and S. P. Wilson: Using DIRECT to Solve an Aircraft Routing Problem. *Computational Optimization and Applications* **21**, 311–323 (2002).

[3] T. Bektas: The multiple salesman problem: an overview of formulations and solution procedures. *Omega* **34**, 209–219 (2006).

[4] T. Bektas, G. Erdogan, and S. Ropke: Formulations and Branch-and-Cut Algorithms for the Generalized Vehicle Routing Problem. *Transportation Science* **45**, 299–316 (2011).

[5] W. M. Carlyle, J. O. Royset, and R. K. Wood: Lagrangian Relaxation and Enumeration for Solving Constrained Shortest-Path Problems. *Networks* **52** (4), 256–270 (2008).

[6] G. Ghiani, and G. Improta: An efficient transformation of the generalized vehicle routing problem. *European Journal of Operational Research* **122**, 11–17 (2000).

[7] K. Helsgaun: An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic. *European Journal of Operational Research*, **126** (1), 106–130 (2000).

[8] G. Laporte, A. Asef-Vazari, and C. Sriskandarajah: Some applications of the generalized traveling salesman problem. *Journal of Operational Research Society* **47**, 1461–1467 (1996).

[9] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnoy Kan, and D. B. Schmoys; The Traveling Salesman Problem. *John Wiley & Sons.* (1985).

[10] C. E. Noon and J. C. Bean: An Efficient Transformation of the Generalized Traveling Salesman Problem. *Technical Report 89-36*, (1989).

[11] C. E. Noon and J. C. Bean: An efficient transformation of the generalized traveling salesman problem. *INFOR* **31**, 39–44 (1993).

[12] J. O. Royset, W. M Carlyle, and R. K. Wood: Routing military aircraft with a constrained shortest-path algorithm. *Military Operations Research* **3**, 31–52 (2009).

[13] M. Zabarankin, S. Uryasev, and R. Murphey: Aircraft Routing under the Risk of Detection. *Wiley Periodicals, Inc.*, 728–747 (2006).

[14] S. Lin and B. W. Kernighan: An Effective Heuristic Algorithm for the Traveling-Salesman Problem. *Operations Research* **21** (2), 498–516 (1973).