FFI-rapport 2008/01317

Resource allocation in military operations – optimization using a genetic algorithm

Maria F. Fauske

Norwegian Defence Research Establishment (FFI)

27 June 2008

FFI-rapport 2008/01317

1068

P: ISBN 978-82-464-1395-2
E: ISBN 978-82-464-1396-0

Keywords

Troops to tasks

Ressursallokering

Prosjektplanlegging

Genetisk algoritme

Militære operasjoner

Approved by

Stein Malerud	Project manager
Espen Skjelland	Director of Research
Jan Erik Torp	Director

English summary

The purpose of this study was to look at methods for resource allocation in military operations. In this report it is shown how a military operation can be viewed as a project, and how methods from project management can be used to automatically generate activity schedules for such operations. Mathematical models of this type are NP-hard, i.e. not solvable within reasonable time. Thus, heuristic algorithms must be used. Such algorithms are fast and flexible. However, solutions found by heuristic algorithms can not be proven to be optimal.

The genetic algorithm was chosen for this study. This algorithm spans out the solution space to a larger extent than other heuristic algorithms. Other advantages of the genetic algorithm are that it seems to be effective as well as robust against getting trapped in local optima.

The genetic algorithm developed in this study was compared to classical optimization methods by solving a very small version of the resource allocation problem. Results showed that the genetic algorithm was able to find an optimal solution and run-time was significantly shorter than for the exact optimization model. The genetic algorithm was used on a larger problem close to what can be found in real life. It was able to find optimal solutions. The solutions will not necessarily be the best in practice, but in an operational context they may be used as a basis for military decision making.

Sammendrag

Hensikten med denne studien var å se på metoder og modeller for ressursallokering i militære operasjoner. I rapporten vises det hvordan en slik operasjon kan sees på som et prosjekt, og hvordan metoder fra prosjektstyring kan benyttes for automatisk å generere aktivitetsplaner. Matematiske modeller av denne typen er "NP-hard". Det vil si at de ikke kan løses innenfor rimelig tid. I slike tilfeller blir ofte heuristiske optimeringsmetoder tatt i bruk. Slike algoritmer er effektive og fleksible, men de kan ikke finne løsninger som beviselig er optimale.

I denne studien ble en genetisk algoritme valgt for å løse ressursallokeringsproblemet. Denne typen algoritme spenner ut løsningsrommet i større grad enn andre heuristiske metoder. I tillegg er den effektiv, og den har gode metoder for å ikke la seg fange i lokale optima.

Algoritmen som ble utviklet gjennom denne studien, ble sammenlignet med eksakt optimering ved at en liten versjon av ressursallokeringsproblemet ble løst ved hjelp av begge metoder. Resultatene viste at den genetiske algoritmen var i stand til å finne optimale løsninger, og at kjøretiden var signifikant kortere enn for den eksakte optimeringsmetoden. Den genetiske algoritmen ble også brukt til å løse et større og mer virkelighetsnært problem. Optimale løsninger ble funnet. Løsningene er ikke nødvendigvis de beste i praksis, men i en operativ sammenheng vil de kunne brukes som utgangspunkt i en militær beslutningsprosess.

Contents

1 Introduction

This report describes the results of a study conducted by project 1068 "Methods and models for analysing peace support and low intensity operations". The purpose of the study was to look at models for resource allocation in military operations. Such operations are often characterized by a large amount of tasks and a limited amount of resources. Today, allocation of resources is done manually which becomes a rather large challenge even with a small amount of activities and resources. This report shows how a military operation can be viewed as a project and how methods from project management can be used to automatically generate optimal activity schedules for such operations. It is also described how to define the problem, how to build the model, and solve it using an heuristic algorithm.

In chapter 2, the military operational planning process is described, and the need for resource allocation in this process is indicated. Also, requirements of a tool for resource allocation are listed. In chapter 3, relevant theory from project management is described. Chapter 4 shows the formulation of a model for military resource allocation based on theory from project management. Limitations of such a formulation are pointed out. Possibilities of extending the model by using heuristic algorithms are described of in chapter 5. It is explained how the problem is solved using a so-called genetic algorithm, and the algorithm itself is described. Chapter 6 shows the results from applying the algorithm to a problem close to what is found in real life. Finally, chapters 7 and 8 contain discussions and conclusions related to the problem and the method used.

2 Resource allocation in military operations

The need for optimal resource allocation in military operations is obvious. Resources are limited and time is short. The next two sections will describe the operational planning process and the requirements of a tool for resource allocation in this context.

2.1 Operational planning process

Planning of military operations follows a strict procedure described in the guidelines for operational planning (GOP) [8]. This operational planning process (OPP) is illustrated in Figure 2.1. The five stages of the process are described briefly in the following.

Stage 1: Initiation
The purpose of stage 1 is to establish the general planning direction, start necessary preperations, collect information and data about the area of operations, and notify relevant personnel.

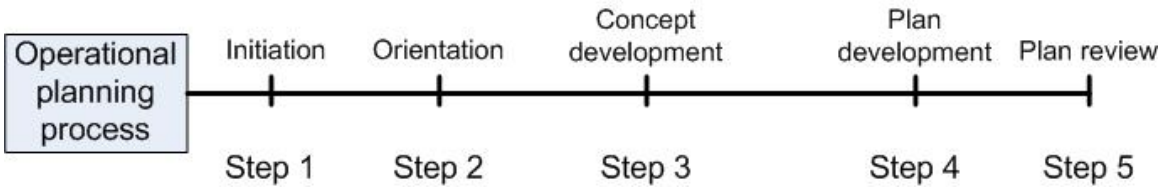


Figure 2.1: Military operational planning process

Stage 2: Orientation
The purpose of stage 2 is to determine what needs to be done to meet the higher authority´s direction and guidance. The goal is to state precisely the mission and desired end-state. Analysis of available resources is performed in this stage.

Stage 3: Concept development
The purpose of stage 3 is to determine how the operation should be carried out to accomplish the mission effectively. A main goal in this stage is the development of courses of action (COA). Troops to tasks analysis is an essential element of this stage.

Stage 4: Plan development
The purpose of this stage is to develop the operational plan.

Stage 5: Plan review
The purpose of this stage is to update the plan to adjust it to recent developments.

One of the main purposes of the OPP is to find the best way of performing the tasks identified to reach the desired end state of the operation. An important element of this is the troops to tasks analysis. A challenge in this analysis is to allocate troops to tasks in such a way that the operation is performed with optimal utilisation of resources. Today, no tool exists for doing this allocation automatically.

As mentioned earlier in this chapter, resource allocation in a military context is not solely relevant for operational planning and troops to tasks analysis. Therefore, in the following the word *resource* will be used instead of *troop*, the word *activity* instead of *task* and the word *project* will be used instead of *operation*. This is more in line with general resource allocation theory within the field of operations research.

In the next section, a requirement specification for a military resource allocation tool is described.

2.2 A tool for automatic resource allocation

The purpose of the tool is to generate a schedule for the project. Available resources should be allocated to activities in an optimal manner. Criteria for optimality may be:

- Minimization of project duration
- Minimization of project costs
- Maximization of project quality
- Finding the schedule that best satisfies activity priorities

These goals can often not be met at the same time, so it may be necessary to choose which goal is the most important.

The tool may also be used to check whether there are enough resources to complete the project within a specified time limit. Such a gap analysis will also tell which types of resources are in shortage. Exploring the consequences of not getting enough resources may also be done.

The allocation of resources will be subject to a large set of requirements and assumptions. These are listed below:

Resources

- Resources may be of one or several resource types
- Activities require either one or more resource types or specific resources
- In the operation plan it has to be specified the exact resources allocated to each activity, not only resource type
- A resource may perform several activities at the same time, as long as its capacity allows it
- A resource´s primary function will be to perform certain activities. However, it may also perform other activities (secondary function) if no other resources are available
- Resources may need resting time after finishing activities
- There may be a hierarchic structure of the resources, i.e. some resources are part of other resources (e.g. a platoon is part of a company)
- Activities may need resources from different levels of the hierarchy

Activities

- Activities have an estimated duration
- Activities may need to be performed at specific locations
- An activity can not be performed before required resources are available
- An activity can not be performed before all precedence activities are finished
- An activity can not be performed before all parallell activities can also be performed
- There may be priorities associated with the activities. High priority activities should be performed first

- An activity may require to start at a specific moment in time
- An activity´s resource requirement may be "as many as possible"

Project

- The project may be divided into phases
- Activities may be required to start within certain phases
- Activities may be required to end within certain phases
- Some activities may be in progress during the whole operation

The implementation of a tool as described above needs to be based on a resource allocation model. Such a model would not be unlike what is found in general project management. Therefore, relevant project scheduling theory is described in the next chapter.

3 Resource-constrained project scheduling

In project management, one of the main tasks is the scheduling of activities. Projects can be found in many areas and due to the diversity in project characteristics and constraints, a great many methods for project scheduling have been subject of research over the years. In general, all project scheduling models consist of three basic components:

- Resources
- Activities
- Precedence relations between activities

This report is concerned with projects where resources are limited and renewable. Such problems are often called *Resource-constrained project scheduling problems* (RCPSP) in the literature, and have been studied within the field of operations research since the mid 1950 [9]. The basic RCPSP consists of the ingredients listed above, and is further defined through constraints related to resource availability and activity durations. The goal of the RCPSP is to minimize total project duration.

The RCPSP can be formulated as a mathematical optimization model. For an introduction to such models, see [5]. The general formulation of the RCPSP is given in e.g. [7], and is shown in the following. Consider a project with J activities, labeled $j = 1, ..., J$. The time horizon of the project is T. The duration of each activity is denoted as d_j. Predecessor activities of activity j are given in the set P_j. There are K renewable resources, labeled $k = 1, ..., K$. Each resource k has an availability of R_k in each time period. Activity j requires r_{jk} units of resource k in each period it is processed. Two additional activities $j = 0$ and $j = J + 1$ represent the start and end of the project. These dummy ativities have 0 duration and do not require any resources. In the mathematical model, decision variables are:

$$x_{jt} = \begin{cases} 1, & \text{if activity } j \text{ is finished at time } t \\ 0, & \text{otherwise} \end{cases} \tag{3.1}$$

The objective of minimizing project duration is equivalent to minimizing the finishing time of dummy activity $J+1$. This can be formulated as:

$$\text{minimize} \qquad \sum_{t=0}^{T} t x_{(J+1)t} \tag{3.2}$$

The model has four constraints:

$$\sum_{t=0}^{T} x_{jt} = 1 \qquad j = 1, ..., J \tag{3.3}$$

$$\sum_{t=0}^{T} t x_{jt} \geq d_i + \sum_{t=0}^{T} t x_{it} \qquad j = \{1, ..., J\}, i \in P_j \tag{3.4}$$

$$\sum_{j=1}^{J} \sum_{u=t}^{t+d_j-1} r_{jk} x_{ju} \leq R_k \qquad k = \{1, ..., K\}, t = \{0, ..., T\} \tag{3.5}$$

$$x_{jt} \in \{0, 1\} \qquad j = \{1, ..., J\}, t = \{0, ..., T\} \tag{3.6}$$

Constraint (3.3) makes sure that each activity has one and only one starting time. In (3.4), precedence relations between activities are handled; Activity j cannot start until all activities in P_j are finished. Constraint (3.5) says that for each resource, at any time, acitivites using the resource at that time cannot use more of it than its capacity. And finally, binary constraints are given in (3.6).

The RCPSP can be varied and extended to give a multitude of resource allocation problems. Common for all of them, is that they are so-called NP-hard. This means that they can not be solved exactly in polynomial time, except from very small versions of the problems. The NP-hardness of the RCPSP was proven in [2]. For a thorough explanation of NP-hardness, see [6].

When optimization problems can not be solved to an exact optimum, heuristic algorithms can often be applied. A solution found by such an algorithm is feasible, but not necessarily optimal. *Most likely*, the solution is better than many other solutions, and *hopefully* it is closer to optimum than most other solutions.

In Chapter 4, it is shown how resource allocation in military operations can be formulated as a

model based on the RCPSP. It is also described how this mathematical formulation can reflect only a very limited version of the real life problem.

4 A model for military resource allocation

When optimizing real life problems, it is often necessary to make simplifications and priorities to formulate a model that is solvable whithin reasonable time. This will of course have implications on the quality of the solution. Not uncommonly, the complexity of the problem makes it very hard to reach the level of simplicity that is necessary to formulate the problem mathematically. In such cases, heuristic solution methods may be used. Such methods are much more flexible than mathematical optimization models in the sense that complexity is not that often a problem. However, solutions found by heuristic methods are not guaranteed to be optimal.

In military resource allocation problems, finding the optimum is not necessarily crucial. The main purpose of a resource allocation tool, is to generate alternative solutions fast. A good solution will often be good enough. There will always be the need for human quality checking of the solutions anyway.

It has already been mentioned that resource allocation in a military context may be based on the RCPSP described in the previous chapter. Comparing the tool requirements from Section 2.2 to the model formulation in Chapter 3, it is easy to see that the model is too simple to cover all aspects of the problem. However, trying to formulate a simplified version of the problem mathematically may be very useful. There are two main reasons for this. First, it enforces a thorough consideration of different aspects of the problem – what is important and what is not. Second, if heuristic methods are to be applied, an exact optimization model may be used to check the quality and performance of the heuristic algorithm.

4.1 Building a simple model based on the RCPSP

4.1.1 Decision variables

In the RCPSP, it is predefined exactly which resources are needed for each activity and the decision is about finding out at what time to allocate those resources to the activities. In a military context however, there will often be several resources that are perfectly capable of doing each activity. Therefore, the decision will be about both finding out *which* resources to allocate to each activity, and at *what time* the activity should be performed. Decision variables will be as follows:

$$x_{jkt} = \begin{cases} 1, & \text{if activity } j \text{ is performed by resource k and finished at time } t \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

With such a formulation, only one resource can perform each activity. In real life, an activity will often require several resources. This is an aspect that will be considered further in Chapter 5, but at this stage the model will be limited to only considering a one-to-one relationship between activity and resource.

4.1.2 Objective function

In Section 2.2, four criteria of optimality where listed. The first one, minimization of project duration, corresponds to the objective function of the RCPSP and will be considered in this chapter. Formulation of this objective is as follows:

$$\text{minimize} \qquad \sum_{k=1}^{K} \sum_{t=0}^{T} t x_{(J+1)kt} \tag{4.2}$$

where K is the total number of resources.

4.1.3 Constraints

Requirements from the list in Section 2.2 are either included in the model as input data, or as constraints. All constraints from the RCPSP are relevant in this model:

$$\sum_{k=1}^{K} \sum_{t=0}^{T} x_{jkt} = 1 \qquad j = 1, ..., J \tag{4.3}$$

$$\sum_{k=1}^{K} \sum_{t=0}^{T} t x_{jkt} \geq d_i + \sum_{k=1}^{K} \sum_{t=0}^{T} t x_{ikt} \qquad j = \{1, ..., J\}, i \in P_j \tag{4.4}$$

$$\sum_{j=1}^{J} \sum_{u=t}^{t+d_j-1} r_{jk} x_{jku} \leq R_k \qquad k = \{1, ..., K\}, t = \{0, ..., T\} \tag{4.5}$$

$$x_{jkt} \in \{0, 1\} \qquad j = \{1, ..., J\}, k = \{1, ..., K\}, t = \{0, ..., T\} \tag{4.6}$$

In addition, Section 2.2 stated the requirement that an activity can not be performed before all paralell activities may also be performed. This can be formulated as the following constraint:

$$\sum_{k=1}^{K}\sum_{t=0}^{T} tx_{jkt} = \sum_{k=1}^{K}\sum_{t=0}^{T} tx_{ikt} \qquad j = \{1, ..., J\}, i \in Q_j \tag{4.7}$$

where Q_j is the set of all parallell activities of activity j.

4.1.4 Solvability

The model formulation above describes a simplified resource allocation problem. Solving the model with GNU linear programming kit (GLPK) [1], illustrates the NP-hardness of this problem through a very fast increase in run-time, see Table 4.1.

No. resources	No. acitivites	Run-time
3	8	a couple of minutes
3	10	a few hours
3	13	> a working day

Table 4.1: Run-times solving the problem with GNU linear programming kit

It is obvious that trying to develop this model further to include all the requirements from Section 2.2 will not serve any good. The problem needs to be solved using an heuristic algorithm. This will be the subject of the next chapter. The mathematical model formulated above can serve as a help to check the quality of the heuristic algorithm, since small and simplified versions of the problem may be solved in both ways.

5 Extended model and heuristics

5.1 Extending the model

The purpose of heuristic algorithms is to search through the solution space in an intelligent manner. That is, searching for the optimal solution without having to look through all possible solutions. See [5] for a more thorough description of heuristic algorithms.

Heuristic algorithms are often implemented with a programming language like Java, C++ or Matlab. This gives the developer flexibility. Developing a good tool consists of two things: 1) Making a well-functioning algorithm, and 2) Implementing all aspects and requirements of the problem at hand. It is not much point in investing a lot of effort in 2), before the algorithm itself has reached a certain level of quality. In developing and testing the algorithm, it is practical if the problem to solve is not too large and complex. This makes it easier to control the testing process and identify

the consequences of making changes to the algorithm. With a basic and well-functioning algorithm at hand, it will be an iterative process to make the tool complete. Hence, fulfilling the requirements of the model and updating the algorithm need to be done in paralell.

In the following, a first small version of the resource allocation problem will be described. This is used as a basis for the development of an heuristic algorithm. The model is extended to some degree compared to the formulation in Chapter 4. From the list of requirements in chapter 2.2, the following are included in the model:

- Resources may be of one or several resource types
- Activities require either one or more resource types or specific resouces
- In the operation plan it has to be specified the exact resources allocated to each activity, not only resource type
- Activities have an estimated duration
- An activity can not be performed before required resources are available
- An activity can not be performed before all precedence activities are finished
- There may be priorities associated with the activities. High priority activities should be performed first

The objective is still to minimize total project duration.

5.2 Solving the problem with a genetic algorithm

Three of the most common heuristic methods in optimization are tabu search, simulated annealing, and genetic algorithms (GA). All these methods have been used to solve the standard RCPSP. See [12], [3], and [7], for examples of this.

In this study, GA was considered to be the most interesting alternative for solving the problem. One of the main advantages of GAs, is that there is no initial guess of the solution. Other heuristic methods require one single starting point, and the final result is highly dependent on which starting point was chosen. Instead, GAs use a search range which spans out the search area. This makes GAs good at not getting trapped in local optima. In addition, GAs are effective and they are relatively easy to implement.

When it comes to scheduling problems, GA is well suited because the structure of the problem can easily be represented in the algorithm. Also, much work has been done in this field already [7], which the problem of resource allocation in military operations can be based on. Last, the field of genetic algorithms in general is relatively new and it is interesting to explore its potensial.

The idea behind GAs is based on principles from evolution. Taking a "population" of solutions and making it evolve through generations by letting the qualities of the good solutions survive, and

letting the bad solutions die out. For a thorough introduction to genetic algorithms, see [11]. In the following, the basic ideas of GAs are described and it is shown how this was implemented for the resource allocation problem. The algorithm was implemented in Matlab.

5.2.1 Basic genetic algorithm

Most methods called GA have at least the following elements in common: populations of chromosomes or individuals, selection accoring to fitness, crossover to produce new offspring, and random mutation of new offspring [11]. One way of formulating a GA, taken from [7], is as follows:

$G := 1$;
generate initial population POP;
compute fitness for individuals $I \in POP$;
while $G <$ number of generations
    $G := G + 1$;
    produce children CHI from POP by crossover;
    apply mutation to children $I \in CHI$;
    compute fitness for children $I \in CHI$;
    $POP := POP \cup CHI$;
    reduce population POP by means of selection;
end

In the following sections, the different elements of GAs are described. It is shown how these elements were implemented in the GA for military resource allocation (GA MRA).

5.2.2 Individuals and fitness

In [7], different ways of representing the individuals for the RCPSP are examined. It is concluded that the *activity based GA* performes best. There, individuals are represented by precedence feasible activity lists. Given activities A1, A2, A3, A4, and A5, and a precedence relation saying that A5 needs to be finished before A2 can start, examples of precedence feasible activity lists may be:

$$(A3, A5, A1, A4, A2) \text{ and } (A4, A1, A5, A2, A3)$$

In the GA MRA, each individual needs to be composed of both activity sequence and the resources allocated to each activity. Each activity requires one or more resource types. Each resource type is associated with a list describing which resources may function as this type. With three resources, $R1, R2$, and $R3$, and six resource types, $T1, T2, T3, T4, T5$, and $T6$, the relationships between resource type and resources may look like:

$$\begin{pmatrix} T1 & R1 \\ T2 & R2 \\ T3 & R3 \\ T4 & R1, R2 \\ T5 & R2, R3 \\ T6 & R1, R2, R3 \end{pmatrix}$$

Resource type requirements of the activities may be:

$$\begin{pmatrix} A1 & T4, T3 \\ A2 & T6 \\ A3 & T1, T4 \\ A4 & T5 \\ A5 & T2 \end{pmatrix}$$

Given these inputs, an example of a feasible individual is:

$$\begin{pmatrix} A3 & R1, R2 \\ A5 & R2 \\ A1 & R1, R3 \\ A4 & R3 \\ A2 & R1 \end{pmatrix}$$

The GA MRA is based on the basic GA from Section 5.2.1. When generating the initial population, a number of individuals equal to the size of POP is generated randomly. The individuals are computed using serial *schedule generation scheme* (SGS). The serial SGS works in the following manner:

> A dummy source activity is scheduled first, to represent the start of the project. Then the set S of scheduled activities is determined. So is the set E of eligible activities, i.e. those activities whose predecessors are already scheduled. From the set E, one activity is picked randomly. Based on the list of resource types this activity requires, resources are picked randomly and allocated to the activity. Now, the activity is scheduled at the earliest possible starting time, that is, when all predecessor activities are finished and when all resources allocated to the activity are no longer in use by other activities. Then the sets S and E are redetermined. This process is repeated until all activities are scheduled. A dummy sink activity represents the end of the project. All other activities

are set as predecessor activities of the sink activity, so the sink activity´s earlist possible starting time is when all activities of the project are finished.

The goodness of an individual is called fitness. For the GA MRA, fitness is given by the total project duration of the individual.

5.2.3 Crossover

Crossover is the operation that gives new solutions based on the population at hand. In each generation, the population is randomly partitioned into pairs of individuals. From these pairs, two offspring are computed. These offspring look a little bit like their "parents", but they are also different. In [7], several crossover variants are described. Here, the so-called *two-point crossover* will be described, as this is the one that was used in the GA MRA.

Two random integers q_1 and q_2 are drawn, with $1 \leq q_1 < q_2 \leq J$, where J is number of activities. The "daughter" is determined in the following way: The first q_1 activities are taken from the "mother". The next $q_2 - q_1$ are taken from the "father". If an activity from the "father" has already been taken from the "mother", this activity is skipped, and the next one is picked instead. At last, the remaining activities are taken from the "mother". This process is illustrated in Figure 5.1. In this example, $q_1 = 2$ and $q_2 = 4$. The "son" is made from the parent individuals by taking the activities that were not picked for the "daughter".

If the two-point crossover scheme is applied to precedence feasible activity lists, the offspring will also be precedence feasible. This is proven in [7].

5.2.4 Mutation

The purpose of mutation is to create activity lists that could not have been produced by the crossover operator. This helps keep up the variety of the population. In the GA MRA, two types of mutation are present.

Mutation of activities works as follows: With J being the total number of activities, for all positions $i = 1, ..., J-1$, activities j_i and j_{i+1} are exchanged with a probability of $p_{actmutation}$, if the

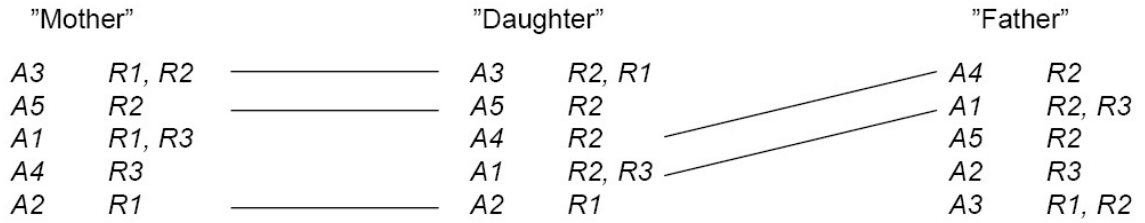


Figure 5.1: Creating a "daughter" from two individuals with $q_1 = 2$ and $q_2 = 4$

result is an activity list which is precedence feasible. This kind of mutation changes the individual, but not necessarily the related schedule. For example, interchanging two activities with the same starting time will give a new individual, but not a new schedule.

Mutation of resources works as follows: For all positions $i = 1, ..., J$ a resource is exchanged with a probability of $p_{resmutation}$. The resource is picked randomly among the resources allocated to the activity. This resource is interchanged with another, randomly picked resource of the correct resource type.

Values of $p_{actmutation}$ and $p_{resmutation}$ are relatively small, typically around $0.05 - 0.10$. It will vary from problem to problem which values will give the best results. Hence, different values should be tested.

5.2.5 Selection

Selection is the process of choosing which individuals will survive and which will die out. In the GA MRA, after crossover and mutation have taken place, the population is twice its initial size. Selection is applied to reduce the population to its former size. Selection concludes each iteration of the algorithm and a new generation is obtained.

There are several ways of deciding which individuals will survice each generation. The one used in the GA MRA is the ranking method. The population is twice its size after the crossover operation, and the half with the worst fitness is simply removed. Another option is to use proportional selection. Individuals are successively picked for the next generation, where the probability of being chosen is proportional to fitness. When the original population size is reached, the individuals that were not picked die out. Third, tournament selection can be applied. Here, individuals are chosen randomly and "compete" for survival. The individual with the best fitness gets a place in the new generation. This process is repeated until the population size is restored.

5.3 Comparison of exact and heuristic solution

Solving a small problem both exactly and with the genetic algorithm indicates the GA´s quality and performance. Input data of the test problem is given in Tables 5.1 and 5.2. There are eight activities and three resources. Some precedence relations exist.

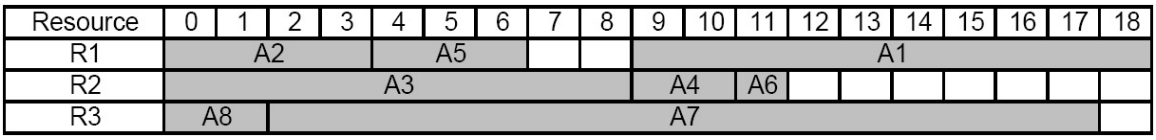
The resulting schedule from the exact optimization is given in Figure 5.2(a). The problem was solved with GLPK, and the run-time of this optimization was 13 minutes. The time line shows that minimal duration is 19 days.

Activity	Duration	Pre. activities	Resource type
A1	10	A3	T4
A2	4		T5
A3	9		T2
A4	2		T6
A5	3	A8	T7
A6	1		T2
A7	16		T3
A8	2		T4

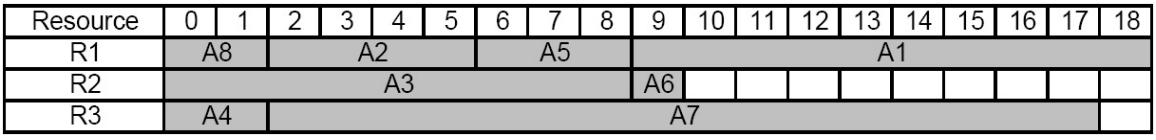
Table 5.1: Input data of small resource allocation problem

Resource type	Resources
T1	R1
T2	R2
T3	R3
T4	R1,R2
T5	R1,R3
T6	R2,R3
T7	R1,R2,R3

Table 5.2: List of resources in each resource type for small resource allocation problem



(a) Classical optimization result



(b) Genetic algorithm result

Figure 5.2: Comparison of results from solving the problem with classical optimization and genetic algorithm

The result from solving the problem with the GA is shown in Figure 5.2(b). The resource allocation is a bit different in this case, but the solution found is optimal. The solution was found after 5 generations, using 50 individuals. Run-time of this optimization was 3 seconds.

Results from this comparison show that the GA is able to find optimal solutions for small problems and that it runs much faster than the exact optimization model. In Chapter 6, the algorithm will be tested on a larger problem.

6 Solving a problem from a real-life context

6.1 Problem definition

The purpose of this chapter is to demonstrate the use of the tool on a problem based on a real-life setting. The case is based on a fictious battle group (BG) in Afghanistan and the resource allocation needs of the commanding officer BG (CO BG). Every week, CO BG and his staff needs to decide which activities should be done and who is to do them. Normally, there are more tasks in the to-do list than available resources are capable to do. Therefore, CO BG needs to prioritize activities. Finding the best resource allocation may be crucial to get as many activities done as possible. Also, it may be useful for CO BG to experiment with different parameters determining the allocation.

When starting a new week, many of the BG´s resources are already tied up in activities. The following resources are available to the CO BG this week:

- Mechanized infantry company x 1
  - Company headquarter x 1
  - Infantry platoon x 3
    - Infantry section x 3
- Armoured engineer reconnaissance party x 1
  - Reconnaissance team x 2
- Headquarter and support company x 1
  - Headquarter platoon x 1
  - Medical platoon x 1
    - Medical section x 3
  - Armour combating platoon x 1
    - Armour combating section x 4
  - Mortar platoon x 1
    - Mortar section x 4
  - Combat service support platoon x 1
    - Combined maintenance team x 3
    - Recovery team x 2
  - Anti-aircraft platoon
    - Anti-aircraft section x 4

As explained in Chapter 5, the tool is not yet capable of considering different levels of resources. Therefore, only the lowest level will be considered here. Those entities are listed below. The entities are assumed to be of approximately the same size – about 8 people.

- Company headquarter (CHQ) x 1
- Infantry section (Inf) x 9
- Reconnaissance team (RT) x 2
- Headquarter platoon (HQ) x 1
- Medical section (Med) x 3
- Armour combating section (AC) x 4
- Mortar section (Mort) x 4
- Combined maintenance team (CM) x 3
- Recovery team (Rec) x 2
- Anti-aircraft section (AA) x 4

CO BG has identified the most important activities for the coming week. He wants to check whether he has enough resources to perform them all within 7 days, and if so, whether some of the resources might have time available to perform some of the many other activities on CO BG´s list which didn´t make the first cut.

Some of the activities are going on 24/7 in the BG while others are single tasks or missions which should be done this exact week. Activities are numbered as an identification, not to show priorities.

1. Operational planning and management (OP&M)
2. Camp security (CS)
3. Quick reaction force (QRF)
4. Medical preparedness and sick quarters (M&SQ)
5. Headquarters management (HQM)
6. Air defence alert (AD)
7. Search and seizure (SS)
8. Check point (CP)
9. Observation point (OP)
10. Social patrol (SP)
11. Road patrol (RP1)
12. Road patrol (RP2)
13. Road patrol (RP3)
14. Road reconnaissance (RR1)
15. Road reconnaissance (RR2)
16. Road reconnaissance (RR3)
17. Road reconnaissance (RR4)
18. Road reconnaissance (RR5)
19. Road reconnaissance (RR6)
20. Road reconnaissance (RR7)
21. Escort of the governor (Esc)
22. Intelligence, surveillance, and reconnaissance (ISR1)
23. Intelligence, surveillance, and reconnaissance (ISR2)

24. Intelligence, surveillance, and reconnaissance (ISR3)
25. Humanitarian support (HS1)
26. Humanitarian support (HS2)
27. Humanitarian support (HS3)

CO BG estimates the duration of each activity and he also identifies precedence relations between the activities. This information is summarized in Table A.1 in Appendix A. It is also necessary that CO BG decides what type of resources each type of activity requires. This is shown in table A.2. Finally, each activity has to be assigned specific resources – not resource types. A matrix showing what resources belong to each resource type is needed. This is shown in table A.3.

6.2 Configuration of the algorithm

The model was run several times with population size varying from 50 to 5000, and number of generations varying between 20 and 50. Several values of the mutation probabilities $p_{actmutation}$ and $p_{resmutation}$ were tested. This did not seem to have any significant implications on the quality of the solutions. Mostly, the values used were 0.05 and 0.1, respectively.

As described earlier, the ranking method was used for selection. In fact, proportional selection was tested at one point. Using this, the run-time of the algorithm was significantly reduced, but the solutions found seemed to be farther from optimality than when ranking selection was used.

6.3 Results

One solution to the problem is illustrated in Figure 6.1. Since some of the activities have a duration of 7 days, we know that total duration can not be shorter than this, thus the solution is optimal in this respect. However, one may argue that the solution found is not the best. For example, maybe it would have been better with a more "compact" schedule. Here, some resources are only occupied for one of the seven days, while others are occupied almost the whole week. If number of resources in use had been minimized, many of the resources would have been free to do other missions or tasks, and the CO BG could add more long lasting activities to his list. It is in any case obvious that CO BG´s first pick of activities was a bit too small, as there is a lot of available time among the resources.

The solution shown in Figure 6.1 was found after 20 generations with a population size of 1000. In Figure 6.2, the development of solutions through the generations is illustrated. The red line shows the worst solution of the generation, and the blue line shows the best. A solution of 8 days was found quite fast, but then it took 18 generation before an optimal solution of 7 days was found.

During testing it varied how fast optimal solutions were found. At one time, optimality was reached

Resource	0	1	2	3	4	5	6	Activities
CHQ	1							1. OP&M
Inf1	15	19	23					15. RR2, 19. RR6, 23. ISR2
Inf2	17							17. RR4
Inf3	3							3. QRF
Inf4	7			22		27		7. SS, 22. ISR1, 27. HS3
Inf5								
Inf6	7	14	16	20				7. SS, 14. RR1, 16. RR3, 20. RR7
Inf7		12						12. RP2
Inf8	7			25		24		7. SS, 25. HS1, 24. ISR3
Inf9	11	13		8			26	11. RP1, 13. RP3, 8. CP, 26. HS2
RT1		13	18	22				13. RP3, 18. RR5, 22. ISR1
RT2	11	12	23			24		11. RP1, 12. RP2, 23. ISR2, 24. ISR3
HQ	5							5. HQM
Med1	4							4. M&SQ
Med2	4							4. M&SQ
Med3	4							4. M&SQ
AC1	9			8			26	9. OP, 8. CP, 26. HS2
AC2								
AC3	2							2. CS
AC4						27		27. HS3
Mort1	7	10						7. SS, 10. SP
Mort2							26	26. HS2
Mort3	3							3. QRF
Mort4								
CM1				25				25. HS1
Cm2								
CM3	21							21. Esc
Rec1	21					27		21. Esc, 27. HS3
Rec2				25				25. HS1
AA1	6							6. AD
AA2			23					23. ISR2
AA3	3							3. QRF
AA4				22		24		22. ISR1, 24. ISR3

Figure 6.1: One example of a schedule with minimal total duration

after 17 generations using only 100 individuals. Another time the algorithm needed 34 generations using 5000 individuals. Many times, the algorithm could not reach solutions better than 8 or 9 days.

Varying mutation probabilities, population size, and number of generations, did not show a pattern in which values were most successful. Of course, increasing population size increases the probability of finding good solutions, but this also increases run-time rather drastically. In addition to finding suiting parameter values, the algorithm can be improved through changing selection and crossover schemes. These issues will be discussed further in Chapter 7.

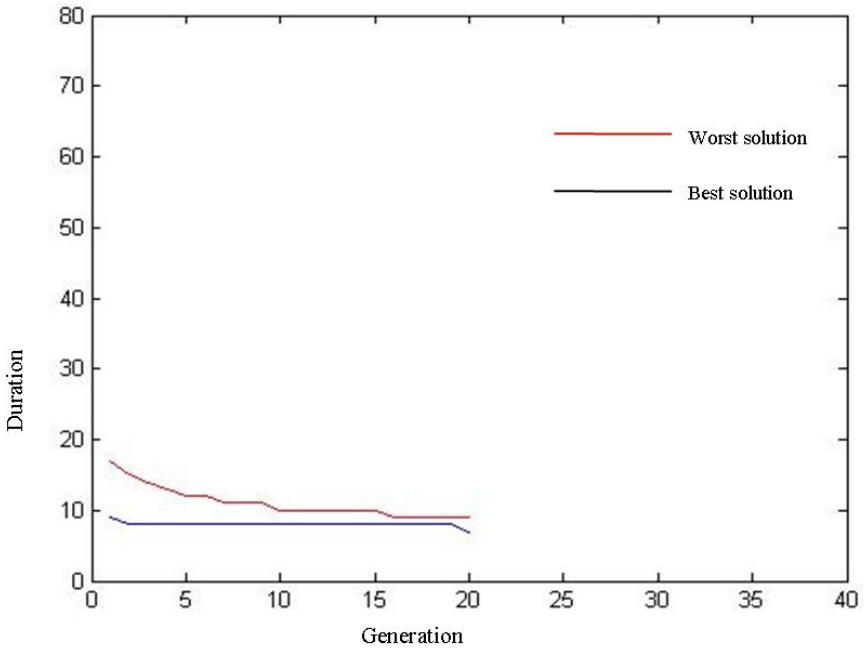


Figure 6.2: Solution development through the generations

7 Discussion

7.1 Purpose of the tool

No tool exists that can completely replace military judgement. This will also be the case for a resource allocation tool. It will mainly be an aid for the commander and his staff – a decision support tool. Generating plans fast, gives the opportunity of experimenting, e.g. by looking at the consequences of changing input values. An interesting aspect may be to look at the sensitivity of activity duration.

The tool can also be used to do gap analyses to identify lack of resources. However, the commander is often well aware that he lacks resources, and is more interested in looking at the consequences of this. The tool may then be used to identify how fast, or maybe how well, the operation could be completed with the resources at hand.

7.2 Is genetic algorithms the solution?

Heuristic methods in general are effective. Whether GA is the best method for the military resource allocation problem can not be stated without testing other methods and comparing them to each

Requirement	Comment
A resource may perform several activities at the same time, as long as its capacity allows it	Easy to include
A resource´s primary function will be to perform certain activities. However, it may also perform other activities (secondary function) if no other resourcecs are available	Easy to include
Resources may need resting time after finishing activities	Easy to include (if resting time can be included in activity duration
An activity can not be performed before all parallell activities can also be performed	Easy to include
An activity may require to start at a specific moment in time	Easy to include
The project may be divided into phases	Easy to include (if starting and end time of phases are known in advance)
Activities may be required to start within certain phases	Can be done by setting minimum and maximum starting time of activity
Activities may be required to end within certain phases	Can be done by setting minimum and maximum starting time of activity
Some activities may be in progress during the whole operation	Easy to include
There may be a hierarchic structure of the resources, i.e. some resoures are part of other resources (eg. platoon is part of company)	Difficult to include. Challenges related to defining the availabiliy and capacity of a higher lever resource if one or more of its lower level resources are occupied with activities
Activities may need to be performed at specific locations	Difficult to include. Travelling time to locations depends on where the resources are situated, which is not known in advance since it depends on allocation process.
An activity´s resource requirement may be "as many as possible"	When resources are scarce, such activities may risk never getting any resources unless it is defined a minimum number required

Table 7.1: Requirements from Section 2.2 not yet included in the model

other. This has not been done in this study. However, it has been confirmed that GAs are flexible and easy to work with. One main advantage compared to other heuristic algorithms, is that GAs span out the solution space to a larger extent. They also have good methods for searching intelligently for optimal solutions.

7.3 Improving the model

The model should be extended to include all requirements from Section 2.2. Table 7.1 lists these requirements and desribes some of the issues related to including them in the model. There are probably a lot of other aspects as well that could be included to make a complete and flexible tool for military operational planning.

It should be mentioned that in a model which deals with future events, uncertainties should al-

ways be considered. A model like the one described in this report, should probably be formulated as a stochastic optimization model. It is claimed in e.g. [10], that a stochastic problem formulated as a deterministic model may yield results that are far from correct. It is especially mentioned that the uncertainty of activity durations in projects is essential in scheduling problems. In further developing the military resource allocation model, the stochastic aspects of the problem should be considered.

7.4 Improving the algorithm

By further testing and developing the algorithm, its performance could most certainly be improved. In the following, some of the main possibilities for algorithm improvements are mentioned.

Algorithm operators
As described earlier, there exist several selection schemes. These should be tested and compared, to see which one performes the best for this exact problem. In addition, further testing of input parameter values should be performed. Finding good values of population size and number of generations is important to optimize the run-time of the algorithm. Values of the two mutation probabilities, $p_{actmutation}$ and $p_{resmutation}$ are also important.

Due to the redundancy properties of resource allocation problems, the importance of $p_{actmutation}$ is reduced. Redundancy is due to the fact that several individuals may have the same related schedule. Thus, performing an activity mutation on an individual may not necessarily change the related schedule. For example, interchanging two activities in the individual which have the same starting time changes the individual, but not the schedule. Since the purpose of mutation is to keep up the variability of the population, redundancy causes the solutions to converge faster.

Convergence
Convergence may be a problem in genetic algorithms. This means that variability of the population is not maintained through the generations. Mutations is one of the things that helps prevent convergence. Another thing that may be effective in this aspect, is multiobjective optimization. Two optimization goals may be defined. Solutions that are good in view of one of the goals, should ideally be bad solutions in view of the other goal. In this way, a larger specter of solutions will live on through the generations, keeping up the variability. In the GA MRA, a relevant goal in addition to minimization of duration, is minimization of number of resources. The use of multiobjective optimization in GA and other evolutionary algorithms is the subject of [4].

Local search
When the algorithm has not improved the best solution for a number of generations, it may be concluded that the algorithm has found a population of good solutions and local search can be applied to improve the solutions that have survived. It is expected that good solutions have similar project

plans and exploring the neighbourhood of good solutions found so far, may lead to even better solutions. This local search process is described further in [7].

8 Conclusions

The purpose of this study was to look at methods for resource allocation in military operations. It was shown that this can be based on resource allocation models from the field of project management. Mathematical models of this type are NP-hard, i.e. not solvable within reasonable time. Thus, heuristic algorithms must be used. Such algorithms are fast and flexible. However, solutions found by heuristic algorithms can not be proven to be optimal.

The genetic algorithm was chosen for this study. This method is based on principles from evolution, where a "population" of solutions are evolved through generations, letting bad solutions die out and good ones live on. Having a population of solutions makes the algorithm span out the solutions space to a larger extent than other heuristic algorithms. Other advantages of the genetic algorithm are that it seems to be effective as well as robust against getting trapped in local optima. However, a comparison with other heuristics has not been made in this study, so it can not be concluded that the genetic algorithm is the best for solving the military resource allocation problem.

The genetic algorithm developed in this study was compared to classical optimization methods by solving a very small version of the resource allocation problem. Results showed that the genetic algorithm was able to find an optimal solution and run-time was significantly shorter than for the exact optimization model. The genetic algorithm was used on a larger problem close to what can be found in real life. It was able to find optimal solutions, but it was obvious that the solution would not necessarily be the best in practice. It can easily be concluded that an automatic tool can not completely replace military judgement. The purpose of such a tool will probably be to use it as decision support for the commanding officer and his staff. The possibility of generating plans fast, gives the opportunity of experimenting with input values and testing different solutions. The tool can also be used to identify consequences of not having enough resources.

In this study, uncertainties and the stochastic properties of the problem were not considered. In further work with models and methods for military resource allocation, these aspect should be studied. The genetic algorithm can also be improved by testing it and its settings more thoroughly than what has been done in this study. It is recommended that the algorithm is improved by testing algorithm operators, finding ways to better handle convergence, and adding a local search phase to the algorithm.

References

[1] GNU linear programming kit: http://www.gnu.org/software/glpk/. *GNU*, 2006.

[2] J. Blazewics, J.K. Lenstra, and A.H.G. Rinnooy Kan. Scheduling subject to resource constraints: Classification and complexity. *Discrete Applied Mathematics*, 5:11–24, 1983.

[3] K. Bouleimen and H. Lecocq. A new efficient simulated annealing algorithm for the resource constrained project scheduling problem and its multiple mode version. *European journal of operational research*, 149(2):268–281, 2003.

[4] K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & sons, LTD, 2004.

[5] M. Fauske. Optimeringsmetoder innen operasjonsanalyse – en oversiktsstudie. FFI-rapport 2008/00123, Forsvarets forskningsinstitutt, 2008.

[6] M.R. Garey and D.S. Johnson. *Computers and intractability – A guide to the Theory of NP-Completeness*. W.H. Freeman and company, 1979.

[7] S. Hartmann. *Project scheduling under limited resources*. Springer-Verlag, 2008.

[8] Supreme headquarters allied power Europe Belgium. *Guidelines for operational planning (GOP)*. NATO, 2005.

[9] W. Herroelen, B. De Reyck, and E. Demeulemeester. Resource-constrained project scheduling: A survey of recent developments. *Computers Ops Res*, 25(4):279–302, 1998.

[10] K. Jörnsten, S. Storøy, and S. W. Wallace. *Operasjonsanalyse*. Cappelen Akademisk Forlag, 1999.

[11] M. Mitchell. *An introduction to genetic algorithms*. MIT Press, 1998.

[12] P.R.Thomas and S. Salhi. A tabu search approach for the resource constrained project scheduling problem. *Journal of Heuristics*, 4:123–139, 1998.

Abbreviations

AC	Armour combating section
AD	Air defence alert
BG	Battle group
CHQ	Company head quarter
CM	Combined maintenance team
COA	Courses of action
CO BG	Commanding officer battle group
CP	Check point
CS	Camp security
Esc	Escort
GA	Genetic algorithm
GA MRA	Genetic algorithm for military resource allocation
GLPK	Gnu linear programming kit
GOP	Guidelines for operational planning
HQ	Head quarter platoon
HQM	Head quarters management
HS	Human support
Inf	Infantry section
ISR	Intelligence, surveillance, and reconnaissance
Med	Medical section
Mort	Mortar section
M&SC	Medical preparedness and sick quarters
OP	Observation point
OP&M	Operational planning and management
OPP	Operational planning process
PRT	Provincial reconstruction team
QRF	Quick reaction force
RCPSP	Resource constrained project scheduling problem
Rec	Recovery team
RP	Road patrol
RR	Road reconnaissance
RT	Reconnaissance team
SP	Social patrol
SS	Search and seizure

Appendix A Input data of case

Activities	Duration	Predecessor activities
1. OP&M	7	
2. CS	7	
3. QRF	7	
4. M&SQ	7	
5. HQM	7	
6. AD	7	
7. SS	1	
8. CP	3	14. RR1
9. OP	3	
10. SP	2	
11. RP1	1	
12. RP2	1	
13. RP3	1	
14. RR1	1	
15. RR2	1	
16. RR3	1	
17. RR4	1	
18. RR5	1	
19. RR6	1	
20. RR7	1	
21. Esc	1	
22. ISR1	2	
23. ISR2	3	
24. ISR3	2	
25. HS1	1	18. RR5
26. HS2	1	19. RR6
27. HS3	1	20. RR7

Table Appendix A.1: Activities with durations (in days) and precedence relations

Activities	Res. types CHQ	Inf	RT	HQ	Med	Mort	AA	General resource	Gen. resource ex. medical.	Search support	Check/obs. point	Road rec.
OP&M	x											
SS		x,x,x								x		
CS									x			
CP											x,x	
OP											x	
SP								x				
RP		x	x									
RR												x
QRF		x				x	x					
Med					x,x,x							
HQM				x								
Esc								x,x				
AA							x					
ISR		x	x				x					
HS								x,x,x				

Table Appendix A.2: Activities´ resource type requirements

	Resource types									
Resources	CHQ	Inf	RT	HQ	Med	AC	Mort	CM	Rec	AA
CHQ	x									
Inf 1-9		x								
RT 1-2			x							
HQ				x						
Med 1-3					x					
AC 1-4						x				
Mort 1-4							x			
CM 1-3								x		
Rec 1-2									x	
AA 1-4										x

	Resource types				
Resources	General	Gen. ex. Med.	Search support	CP/OP	Road rec.
CHQ	x	x			
Inf 1-9	x	x		x	x
RT 1-2	x	x		x	x
HQ	x	x			
Med 1-3	x				
AC 1-4	x	x	x	x	
Mort 1-4	x	x	x	x	
CM 1-3	x	x			
Rec 1-2	x	x			
AA 1-4	x	x		x	

Table Appendix A.3: Resource types and available resources