Université de Tunis
Institut Supérieur de Gestion
LARODEC Laboratory

Evolutionary Resolution Approaches for the Multi-Objective & Multi-Mode Assignment and Scheduling problem

Elaborated by: Olfa Dridi Ben Zekri

Co-advised by: Saoussen Krichen
Adel Guitouni

Abstract

Multi-Objective & Multi-Mode Assignment and Scheduling problem

Contents

Abstract **2**

Introduction **11**

I State of the Art **14**

1 Project Scheduling Problems **15**

1.1 Introduction 15

1.2 Classification of the scheduling problems 16

1.2.1 Resource environment 16

1.2.2 Activity characteristics 17

1.2.3 Optimality criterion 18

1.3 Single-mode Resource Constrained Project Scheduling Problems (SR-CPSP) 19

1.3.1 Basic RCPSP 20

1.3.2 Preemptive RCPSP 20

1.3.3 Generalized precedence relations RCPSP 21

1.3.4 General RCPSP 21

1.4 Multi-mode Resource Constrained Project Scheduling Problems (MR-CPSP) 22

1.4.1 Non-renewable Resource MRCPSP 23

1.4.2 Renewable Resource MRCPSP 23

1.4.3 General MRCPSP 24
1.5 RCPSP variants 24
1.5.1 Stochastic RCPSP 24
1.5.2 Dynamic RCPSP 24
1.5.3 Bin-packing related RCPSP 25
1.5.4 Multi-RCPSP 25
1.5.5 Multi-project RCPSP 26
1.5.6 RCPSP with non-regular objective functions 26
1.6 Problem complexity 27
1.7 Problem settings 27
1.7.1 Definition 29
1.7.2 Environment description 29
1.7.3 The model 31
1.8 Past researches on ASPs 32
1.8.1 Exact methods 32
Linear programming based techniques 33
Branch & Bound procedures (B&B) 33
Dynamic programming (DP) 34
1.8.2 Approximation approaches 36
Full space optimization methods 36
Decomposition methods: Bi-level optimization approaches . . 38
1.9 Conclusion 41

2 Population Based Optimization Approaches 42
2.1 Introduction 42
2.2 Evolutionary algorithms 43
2.2.1 Genetic algorithm 45
2.2.2 Multi-objective evolutionary algorithms 47
2.3 The ant colony optimization algorithms 49
2.4 Particale swarm optimization 51
2.5 Memetic algorithms 53

2.6 Conclusion 53

II Contributions 55

3 A Genetic Local Search Approach for the Single-mode Resource Allocation and Scheduling Problem 56

3.1 Introduction 56
3.2 Model description and formulation 58
3.3 Model implementation 60
3.4 Experimental analysis 63
3.4.1 Parameter settings 64
3.4.2 Performance evaluation 64
3.5 Conclusion 67

4 A Genetic Approach for the Multi-Objective and Multi-mode Resource Allocation and Scheduling Problem 68

4.1 Introduction 68
4.2 Problem description 69
4.3 Problem formulation 71
4.4 Review of the related work 73
4.5 Genetic Algorithm: background 77
4.6 A MCGA for the MOMRASP 78
4.6.1 Encoding variables 78
Encoding of the mode 79
Encoding of the period 79
Encoding of the priority 79
Fitness functions 80
4.6.2 Evolutionary operators 80
4.6.3 The algorithm 82
4.7 Computational experiments 83
4.7.1 Parameter settings 84

4.7.2 Experimental results 85
4.7.3 Diversification of the approximation set based on the diversity indicators 87
4.8 Conclusion 89

5 An Evolutionary Algorithm for the Bi-level Multi-platform Large Volume Surveillance Problem 94
5.1 Introduction 94
5.2 Problem description 96
5.3 Mixed-integer linear formulation 97
5.3.1 Notations 97
5.3.2 Bi-level formulation 99
Upper level formulation 99
Lower level formulation 100
5.4 Solution approach 101
5.4.1 Basic concept and solution framework 101
5.4.2 Construction approach: ACO 102
5.4.3 Upper level search strategy - C_{max} minimization 107
5.4.4 Lower level search strategy - Cost minimization 108
5.5 Empirical results 108
5.6 Conclusion 108

General conclusion 110

Bibliography 111

List of Figures

1.1 Large scale surveillance problem 30

2.1 Crossover and mutation operators 47

4.1 An illustrative example with $N = 6$ and $k = 4$ 70
4.2 The crossover operator 81
4.3 The algorithm 83
4.4 The CPU time for problems P_1-P_{17} 86
4.5 Diversity of the pareto approximation front for problems P_1-P_{17} . . . 90

5.1 Bi-level approach 102
5.2 Graph representation of an ASP instance 105

List of Tables

3.1 Symbols for the mathematical model (1)-(6) 59
3.2 An RCPSP with $n = 4$ and $|R| = 1$ 60
3.3 Meta-tuning of the glsa 64
3.4 Average deviations from the optimal solutions of $J30$ 65
3.5 Average deviations from critical-path based lower bounds of $J60$ 66

4.1 Notations 72
4.2 The configuration of the parameters 84
4.3 Algorithm performances with C_1, C_2 and C_3 85
4.4 Test results for different problem sizes 92
4.5 Computational results of the $MCGA$ and AS appraoches 93
4.6 Comparison between MCGA and AS approaches based on Cov_{F_i} metric for P_1-P_{17} 93
4.7 p-values of the MCGA vs AS comparisons 93

5.1 Computational results 109

Acknowledgements

I am deeply indebted to my supervisors Doctor Saoussen Krichen and Professor Adel Guitouni for their support, encouragement and valuable discussions concerning this research. I greatly benefited from their guidance during this work.
I wish to thank all person who provided me information, advice, criticism and encouragement.

Dedication

To the memory of my Grand Mother,

To my dear parents,

To my husband Riadh,

To my sons Mohamed Anas & Ahmed Firas ,

To my sisters,

To all my family and my friends.

Olfa Dridi Ben Zekri

Introduction

Project management dates back to five hundred years. Historically, until 1900, massively engineered and extremely complex projects are managed using only creativity of constructors and rudimential and informal tools. Among these realisations, we can cite the Egyptian pyramids as well as the Great Wall in china.

Project scheduling has been considered as one of the most challenging issues in project management during the last five decades. In the 1950s, it was originated as a research area with some publications due to pioneers Johnson [53], Jackson [50] and Smith [95]. Over 1960s, integer programming formulations and dynamic programming have received considerable attention as useful tools in solving scheduling problems. During 1970s, after the famous paper of Karp [55] on complexity theory, the complexity hierarchy of scheduling problems is considered. In the early 1980s, scheduling theory came under serious scrutiny. Since then, the field has garnered a lot of researcher's attention in the domains of manufacturing, transportation, engineering and constructions.

Project scheduling concerns the assignment of resources to a set of tasks over time in order to optimize one or more objectives and to satisfy a set of constraints. The Manhattan Project which give rise to the first atomic bomb and Polaris Missile project conceived to create and launch nuclear missiles represent the most famous large scale projects that allowed to the impetus of the management strategies and tools. Nowadays, the competitive environment and the globalization rules dictate that only the best who has the most sophisticated and advanced strategies of production and management can survive. As a result, project scheduling and resource management solutions are in demand throughout the world as a fundamental tool

for the survival and success of companies. It is then the raison for the considerable efforts dedicated to the project scheduling and resource management problem, giving place to a large variety of models and formulations that can be classified as exact and heuristics methods. Generally, exact methods are able to solve at optimality only small or moderate sized problems. Consequently, heuristic approaches are well adopted to find efficient solutions for extended sized problems within a reasonable praticable time. Even more, the real-world decision problems usually involve several objectives that should be simultaneously considered and optimized.

In this thesis, our interest is focused on developing heuristic methods for variants of assignment and scheduling problem (ASP). A first goal of this work is the investigation of the multi-objectif and multi-mode assignment and scheduling problem for large volume surveillance applications. We consider three objectives functions to be optimized simultaneously which are the makespan minimization, the renewable resource cost minimization and the probability of success maximization.

The second goal is the bi-level multi-mode assignment and scheduling problem. We propose an alternative formulation for the problem as a bi-level ASP in order to make easier when adding more realistic assumptions as autonomy (fuel constraints) and travel time.

Finally, as an application to our proposed models, we choose the surveillance problem using a real simulation environment called Inform Lab (IL). It is a testbed to support the development of two groups of algorithms, which are particularly useful for wide-area surveillance applications: Distributed Dynamic Information Fusion (DIF) and Distributed Dynamic Resource Management (DRM). It was initiated by DRDC, but is not restricted to the military domain. In our use case, needs for efficient management solutions is investigated to optimize the assets assignment and system scheduling for surveillance applications. Surveillance assets can be mobile (such as helicopters, ships, airborne, etc) or fixed (as land radar, satellite, etc). the surveillance problem can be stated as a set of heterogeneous and networked resources/assets that should be used to schedule a set of surveillance and urgent tasks.

Our thesis is composed of four chapters and summerized as follows:

Outline of the thesis

Chapter 1 is devoted to the scheduling problems. Starts by presenting the different classes and variants of the resource constrained project scheduling problems (RCPSP). Next, the computational complexity of scheduling problems is given. Finally, a detailed description of the multi-objective and multi-mode assignment and scheduling optimization problem addressed to the multi-platform large volume surveillance problem including the features and the requirements is presented.

Chapter 2 presents a literature review on the assignment and scheduling problem. Previous works listed in exact and heuristic methods aret provided.

Chapter 3 gives the details of the proposed Multi criteria Genetic Algorithm (MCGA) for solving the resource management and employment to accomplish efficiently a set of surveillance tasks as a multi-objective and multi-mode resource allocation and scheduling problem.

Chapter 4 provides the formulation and the proposed approach for solving the bi-level assignment and scheduling problem with time windows.

Finally, we conclude our thesis by providing a synthesis of our research and indicating some directions for future research.

Part I

State of the Art

Chapter 1

Project Scheduling Problems

1.1 Introduction

Scheduling is a decision-making process that models a wide variety of pratical situations as manufacturing systems, logistic processes in the supply chain, wireless systems and surveillance missions [].

It involves a set of resources to accomplish some tasks over time periods in order to optimize one or several objectives. Resources can designate machines in a workshop or runways at an airport, crews at a construction site or processing units in a computing environment. Tasks can be operations in a production process, take-offs and landings at an airport, stages in a construction project or executions of computer programs. Moreover, in such rough and competitive environment, real world decision problems are extremely complex and often require the consideration of several objectives while satisfying hard constraints. As a result, the main concern of the managers is to deliver sophisticated solutions strategies in an attempt to reach the predefined objectives and fulfill all system constraints. Hence, the recourse to project scheduling and resource assignment and task scheduling problem, which can be defined as an allocation of resources to accomplish tasks, and a scheduling of those tasks onto successive periods.

This chapter covers topics on project scheduling classification, its basic variants, and a study of its complexity.

1.2 Classification of the scheduling problems

In literature, there are essentially two main features: the machine scheduling problems and the project scheduling problems. Machine scheduling problems usually depend on the machines environment and the jobs particularities. Graham et al. [35] introduced the standard $\alpha|\beta|\gamma$ notation for representing the machine scheduling problem that expresses three main fields: the machine environment α that allows for the identification of single machine problems, various types of parallel machines problems, flow shops, job shops, general shops, open shops, mixed shops and multiprocessor task systems. The second field β is used to describe the task and resource characteristics. It includes parameter settings for characterising the possibility of preemption, precedence constraints, ready times, deadlines, task processing times, batching and additional resources. The third field γ denotes the optimality criterion. Later Brucker et al. [14] have provided a notation to classify the project scheduling problems according to the resource configuration and the nature of the tasks. The proposed notation is an extension of the three-field notation $\alpha|\beta|\gamma$ for machine scheduling problems introduced previously by Graham et al. [35]. In the context of project scheduling, the first field α used to describe resources characteristics. To distinguish between machine scheduling problems and project scheduling problems, Brucker et al. [14] have introduced in the α-field PS for project scheduling and MPS to capture multiple modes execution for activities. β used to describe activities as in the machine scheduling and γ denotes the problem's objective.

1.2.1 Resource environment

Resource characteristics are specified in the field α. The possible resource environments specified in the α field are:

- One resource $\alpha = PS1$: One resource type is considered for each task.

- Multiple resources $\alpha = PSm$: It is an extension of the $\alpha = PS1$, where there are m resources type available that can be used to process each task.

- Resource constraints $\alpha = PSm, 1, 1$: There are m resources with 1 unit availability of very one and each activity requires at most 1 unit.

- Resource leveling $\alpha = PSm, \infty$: There are m resources with unlimited number of units available (i.e., there are no explicit resource constraints) used for resolving conflicts or over allocations in the project scheduling.

1.2.2 Activity characteristics

Several possible activity characteristics specified by the second field β may modify the scheduling environment. Some of these characteristics are:

- Preemption $\beta = pmtn$: implies that the processing of an activity may be interrupted and resumed at a later time.

- Precedence constraints $\beta = prec$: precedence constraints requiring that one or more activities may have to be completed before another activity is allowed to start its processing.

- Release dates $\beta = r_i$: activity i cannot start its processing before its release date r_i. If r_i does not appear in the β field, the processing of activity i may start at any time.

- Delivery times $\beta = q_i$: after finishing its processing, each activity has to spend an amount of time before exiting the sytem.

- Deadline for project duration $\beta = \bar{d}$: A deadline that indicates when all project activities should finish.

- Due date $\beta = d_i$: Each activity i is specified by a due date d_i. It may be violated at a specific cost.

- Weight $\beta = w_i$: The weight w_i of activity i denote its importance or priority relative to the other activities in the system.

- Mode Alternatives $\beta = mu$: Activities have multiple prespecified execution modes, otherwise activities must be performed in a single execution mode.

The activity characteristics listed above are static prefixed data, since they do not depend on the schedule. Conversely, data that are not fixed in advance and depend on the schedule are called dynamic data as processing times and task arrivals.

1.2.3 Optimality criterion

The third field γ is reserved to denote the optimality criterion. The most commonly chosen criteria involve the minimization of:

- The makespan (C_{max}): the most used objective in project scheduling problems. It represents the completion time of the last activity of the project.

$$C_{max} = max\{C_i | i = 1, \ldots, n\} \tag{1.1}$$

  Where C_i denote the finishing time of activity i.

- The total cost: denote the associated cost by $f_i(C_i)$. There are essentially two types of total cost functions:

$$f_{max}(C) = max\{f_i(C_i) | i = 1, \ldots, n\} \tag{1.2}$$

  and

$$\sum f_i(C) = \sum_{i=1}^{n} f_i(C_i) \tag{1.3}$$

  called bottleneck objectives and sum objectives, respectively. The scheduling problem is to find a feasible schedule that minimizes the total cost function, we set $\gamma = f_{max}$ or $\gamma = \sum_{i=1}^{n} f_i$.

- Maximum lateness (L_{max}): the worst violation of the due dates. The activity lateness is non-negative if it is completed late and negative otherwise.
- Maximum tardiness (T_{max}): the difference between tardiness and lateness is that tardiness equals zero if the activity is completed early (i.e. $T_{max} = max\{0; L_{max}\}$).
- Maximum flow time (F_{max}): The flow time of an activity denotes the time elapsed from its entry to its exit from the system.
- Total (weighted) completion time ($\sum C_i$ or $\sum w_i C_i$): The sum of the (weighted) completion times. It indicates the total holding (or inventory) costs incurred by the schedule. This criterion is often referred to as the total (weighted) flow time.

1.3 Single-mode Resource Constrained Project Scheduling Problems (SRCPSP)

In the SRCPSP, each task has only one execution mode, this means that the processing time and the requirements for one or several resources are assumed to be fixed in advance and only renewable resources are given. This standard project scheduling problem is denoted by $PS|prec|C_{max}$ based on the notation introduced by Brucker et al. [14]. According to Herroelen et al. [42], the SRCPSP is composed of four main sub-classes: basic RCPSP, preemptive RCPSP, general precedence relations RCPSP and general RCPSP.

In general, two graphical representations, activity-on-arc (AOA) and activity-on-node (AON), have been used to describe the RCPSP as a project networks. In the AOA representation, nodes represent events and arcs represent activities/tasks. The finish-start precedence relationships between two activities/tasks i and j, denoted as $i \prec j$ is represented by having the terminal node of the arc i coincide with the start node of the arc j. In the AON representation, activities and their associated parameter information are represented within the nodes and the precedence relations are represented by

directed arcs. Dummy tasks (nodes) are needed to identify the start and completion of the project. We distinguish two categories of the precedence constraints:

- The jth activity can start at any time following the completion of the ith activity.
- The jth activity must start within some time window following the completion of the ith activity: these restrictions are called General Precedence Relations (GPRs). GPRs represent a minimal or maximal time-lag between a pair of activities.

1.3.1 Basic RCPSP

The classical RCPSP consists in scheduling, without preemption a set of n tasks numbered from 1 to n. The processing time of each task i, $(i = 1, \ldots, n)$ is denoted by p_i. Each task i requires r_{ik} units of resource of type k. The availability of each resource k of the K resource type at each period is R_k units, for $k = 1, \ldots, K$. There are precedence relations between the tasks and a task cannot be started until all its predecessors have been accomplished. Dummy tasks 1 and n represent the source and sink of the project, having null duration and do not require any resource consumption. We can consider a due date for the tasks achievement d_i, for $i = 1, \ldots, n$. Let s_i and f_i denote the start and finish time of task i respectively. The solution for the RCPSP is a feasible schedule with a minimum makespan. Hence, the objective is to find the vector of finishing times of all tasks: $f = \{f_1, f_2, \ldots, f_n\}$.

1.3.2 Preemptive RCPSP

The basic RCPSP assumes that a task cannot be interrupted once started. Conversely, a preemptive RCPSP allows tasks preemption at discrete points in time. A task can be interrupted after each integer unit of its processing time. The processing time p_i of a task i may be split into p_i process units. Each process unit $j = 1, 2, \ldots, p_i$ of task i is then assigned an integer finish time f_{ij}. f_{i0} denotes the latest finish time of all predecessors of task i and then corresponds to the earliest start time of task

i. In the $\alpha|\beta|\gamma$ notation of Brucker et al. [14], preemptive RCPSP is represented by adding the value *pmtn* to β.

1.3.3 Generalized precedence relations RCPSP

In the classical RCPSP, a task should be accomplished before the starting time of all its successors. This basic precedence concept can be extended by the specification of minimal and maximal time lags between tasks i and j. Minimal time lags between the completion time C_i of a task i and the start time S_j of its successor task j, means that task j must start unless its predecessor i has already finished within some time window l_{ij}: $C_i + l_{ij} \leq S_j$. Similarly to minimal time lags, the maximal time lag between tasks i and j, means that the start time of a task j may not be later than $\bar{l}_{ij}$ time after the completion time of its predecessor task i: $C_i + \bar{l}_{ij} \geq S_j$. The interval $[S_i - l_{ij}, S_i + l_{ij}]$ is called the time window of starting task j relative to start time of task i. In the $\alpha|\beta|\gamma$ notation of Brucker et al. [14], generalized temporal constraints is represented by adding the value *temp* to β.
The generalized precedence relations RCPSP is known to be strongly $\mathcal{NP}$-hard, and even testing if an instance has a feasible solution is $\mathcal{NP}$-complete. The first optimal solution was found only by the branch-and-bound procedures.

1.3.4 General RCPSP

The general RCPSP is a problem where all the performance measures are regular. A regular objective function (ROF) is a non-decreasing function of the activity start times. ROFs have received much more attention in the literature than non-regular objective functions, especially the project completion time or makespan, the weighted delay, the maximum tardiness, the total number of tardy activities and the weighted mean flow time.

1.4 Multi-mode Resource Constrained Project Scheduling Problems (MRCPSP)

Instead of working with a single mode determined by a fixed duration and a fixed resource consumption, several possible execution scenarios/modes may be defined, where each one reflecting a feasible way to combine an activity duration and a resource allocation. The multiple modes give rise to several types of trade-off. The MRCPSP are often confused with the discrete trade-off problems. Various types of trade-off problems have long been recognized in the context of project scheduling: discrete time/resource trade-off problem (DTRTP), discrete resource/resource trade-off problem (DRRTP) and discrete time/cost trade-off problem (DTCTP).

The DTRTP arises when the duration of a task is assumed to be discrete and non-increasing function of the amount of a single renewable resource assigned to it. For example, a given task can be achieved by one worker in two days, whereas, two workers can finish the same task in only one day. The DRRTP, occurs if resource substitution is assumed. For example, a given task can be achieved in only one day by two experienced workers, or alternatively, by one experienced worker and two apprentices within the same duration. The DTCTP assumes a single non-renewable resource and arises when certain task durations can be reduced at the expense of higher direct cost. The project budget can be regarded as the resource to be allocated. In this context, the MRCPSP can includes time/resource, resource/resource and time/cost trade-offs, multiple renewable and non-renewable resources and a variety of objective functions.

In the basic problem setting, a task i have to be performed in one of their possible execution modes $m_i \in \{1, \ldots, M_i\}$, with M_i the number of modes of i. Each mode implies a different options in terms of cost c_{m_i}, processing time p_{im_i} and amount of renewable and non-renewable resources for achieving the task.

1.4.1 Non-renewable Resource MRCPSP

The availability of non-renewable resources, does not refer to individual points in time but to the entire planning period. Each time an activity is carried out, the residual availability of a non-renewable resource is decreased by the corresponding resource demand. The project budget can serve as example. The MRCPSP with non-renewable resource can be regarded as a linear or discrete time/cost trade-off problem. In the linear time/cost trade-off, activities durations are bounded, without reference to the project completion deadline, a manager would choose a duration which implies minimum direct cost, called the normal duration. At the other extreme, a manager might choose to complete the activity in the minimum possible time, called crashed duration, but at a maximum cost.

The DTCTP, the set of mode is represented as integers and an activity assumes different execution modes according to the possible resource allocations. The problem consists to decide on the vector of activity durations that completes the project as early as possible under the limited availability of the nonrenewable resource.

Kelly [56] formulated the time-cost trade-off problems using linear programming by assuming a linear relationship between time and cost for construction activities. This approach was limited to two options per activity, with a continuous linear variation between them. Meyer and Shaffer [89] used integer programming to solve time-cost problems including both linear and discrete relationships within the same activity. However, the solution process is time consuming.

1.4.2 Renewable Resource MRCPSP

Renewable resources are recoverable after serving a task with its full capacity during the planning horizon. Only the total amount of resources use at every time instant is constrained. The renewable resource MRCPSP can be assimilated to the DTRTP when the duration of a given task is considered as a discrete and non-increasing function of the amount of the renewable resource allocated to it. Manpower, machines, tools and equipment can serve as examples.

1.4.3 General MRCPSP

A regular measure of performance is considered. Activity duration can often vary depending upon the type and amount of resources that are applied i.e. shortening the duration on an activity will normally increase its direct cost which comprises. Regular measures of performance are also referred to as early completion measures. Obviously, minimising the project makespan is a regular performance measure.

1.5 RCPSP variants

1.5.1 Stochastic RCPSP

In order to cope with uncertainty, activity duration and resource requirement are considered as random variables following probability distribution. In this case, instead of minimizing the makespan, we consider the minimization of the expected makespn. Many interdependent activities are often represented in the project with hightly interdependent in the completion times. The probability distribution of the total makespan is sometimes difficult or impossible to define and often leading to activities independent assumptions for tractable analysis. Given the presence of both resource constraints and random activity durations, schedules are generated through the application of so-called scheduling policies or scheduling strategies.

1.5.2 Dynamic RCPSP

The dynamic RCPSP consists to search efficient schedules under a set of constraints and the presence of inevitable real-time events (weather delays, unavailability of resources, delays in the predecessors of an activity, etc) that may affect the performance of scheduling process. Two main classical strategies are used:

❶ Rescheduling strategy: regenerate a new schedule when unexpected events disrupt the initial one. It can generate and maintain optimal solutions, but in practice is rarelly achievable and requires additional computation time and cost.

❷ Repairing strategy: consists to adjust the current schedule when unexpected events disrupt the original schedule. This strategy is simple and aimed to a quick schedule restoriation due to the potential saving in CPU time and the stability of the system.

1.5.3 Bin-packing related RCPSP

A bin-packing problem can be described as follows: given a standard bin size and a set of items, the objective is to assign each item into a bin while minimizing the total number of bins used. A bin-packing-related RCPSP refers to a special kind of resource-constrained project scheduling problem that can be seen as analogous to a bin-packing problem. Many resource constrained project scheduling problems can be viewed as a generalization of the basic bin-packing problem. The analogy between the bin packing and RCPS problems can be explained as follows: The resource availability represents the bin size, while a tasks resource consumption needs represent an item size. In the RCPSP, we can modelize each period as a bin into which we can pack different tasks. We can also make the following assumptions: each task takes less than one period of resource consumption and every task must be fully completed within a single day, so, minimizing makespan of this RCPSP is equivalent to minimizing the total number of bins used in an equivalent bin-packing problem.

1.5.4 Multi-RCPSP

In the multi-resource-constrained project scheduling problem (M-RCPSP), a job may require a set of operations, or a set of resources. For a given operation, various resources can be used in parallel. In other words, to be executed each task/job might select any one of these resources. Where successive resources are needed in series and if a task select a given resource then it is not allowed to use another resource before completing processing on the first one. These problems are often called machine-scheduling problems. In the manufacturing area, a resource is equivalent to a machine that can only process one job at any time. Machine and job scheduling characteristics can be classified to several categories:

1. Identical machines in parallel with the same speeds. Thus, the processing time of job i on machine j is $p_{ij} = p_i$ for all machines j.

2. Machines in parallel with different speeds: the speed of machine i is v_i for all jobs. The processing time of job i depend on the speed v_j of machine j, $p_{ij} = \frac{p_i}{v_j}$.

3. Unrelated machines in parallel: where the machines have different speeds which are task-dependent, $p_{ij} = \frac{p_i}{v_{ij}}$.

4. Flow shop: there are m machines in series and each job has to be processed on each of the m machines. All jobs have to be processed on each of the machines in the same sequence/routing, that is, they have to be processed first on machine 1, then on machine 2, and so on. The number of machines in the flow shop is a constant, the value of which is specified as part of the problem type.

5. Job shop: is similar to the flow shop, with the only difference that each job has its own route to follow.

1.5.5 Multi-project RCPSP

The multi-project RCPSP is a generalization of RCPSP. It consists of a set of I projects to be scheduled simultaneously, with $I > 1$, where each one is composed of n activities. All these projects share some or all of the available resources.

1.5.6 RCPSP with non-regular objective functions

As defined above, the regular objective function is one in which the objective function is never made worst by reducing the task duration without increasing the duration of any other task. However, the non-regular objective function don't respect this property, because the value of the objective function can actually increase by reducing task duration.

1.6 Problem complexity

Complexity theory provides a mathematical framework in which computational problems can be classified as polynomial ($\mathcal{P}$) or non polynomial ($\mathcal{NP}$). The evaluation of an algorithm complexity consists to find a function f that maps each input x in some given domain to an output $f(x)$ in some given range. Complexity theory is interested in studying the time required by an algorithm to compute an output $f(x)$. A problem is called a decision problem if the output range is $\{yes, no\}$. The complexity class $\mathcal{P}$ consists of all decision problems that may be solved in a polynomial time. The $\mathcal{NP}$ class consists of all decision problems for which no polynomial time algorithms are known but for which the *yes* answer can be verified in polynomial time. The RCPSP is an optimization problem that, consists in finding the best solution from all feasible solutions, can be formulated as a decision problem. The knowledge that the scheduling problems are $\mathcal{NP}$-hard and as a generalization of the job-shop problem, the RCPSP appertains to the class of $\mathcal{NP}$-hard optimization problems in the strong sense [9]. Meaning that the computing time grows exponentially with problem size in the worst case. Besides, there is no available algorithm that can find the optimal solution exactly within polynomial time only for small-sized problems. Another possibility is to apply approximation methods. These methods are widely used in practice to solve large scale problem instances in order to cope with complex, highly combinatorial sequencing and scheduling problems.

1.7 Problem settings

The interest to the resource management and project scheduling subject during the last decades was motivated by the advantage of the development in computer technology, telecommunication, web application, as well as marketing strategies and also the influence of the growth number of competitors. Hence, the manager is concerned to deliver enough innovative and sophisticated solutions strategies in an attempt to promote development and avoid failure. For example, an efficient management of all levels of the supply chain is a primodial element of successful companies.

An other typical example, the surveillance problem. It is an extremely complex problem requiring the consideration of several objectives and constraints, and is widely used in military domain, target tracking, battlefield [81] [5]. We can mentionned the surveillance of the land borders territory, viz, the coastal surveillance problem, which represents a major problem for many countries, from many variants of threats and illegal activities, as unauthorized vessel traffics [81].

The large volume surveillance problem is generally characterized by the employment of mobile (maritime patrol aircraft, helicopters, UAVs, ships) and fixed surveillance assets (e.g. land radar) to a large geographic area in order to identify, assess and track the maximum number of moving, stopped or drifting entities and objects. The observed objects are not necessarily aware of being observed and are cooperative or non-cooperative, and friendly or hostile. Coastal and Arctic Wide Area surveillance are good examples of large volume surveillance. The scarce surveillance (e.g. Electro-Optical (EO), Infra-Red (IR), and Synthetic Aperture Radar (SAR) sensors) and tracking capabilities (normal radar modes) makes it very difficult to perform large volume surveillance and to keep track of all activities. Therefore, one of the decision problem of the large volume surveillance is the determination of employment and management solutions of surveillance and patrolling assets to accomplish efficiently a set of surveillance tasks. Surveillance tasks are characterized by their dependencies, priorities, as well as time and space constraints. Moreover, tasks can be performed according to several modes (i.e., depending on the assets assigned to the task). Each mode can be characterized by a task-time duration, resource consumption, cost and benefits (e.g., probability of achieving successfully the task). Each mode is generally executed by one or a set of surveillance assets, which have different capabilities, cost, benefits, etc. Mobile surveillance assets require generally preparation, transit and sometimes transit times. This problem can be stated as a variant of the multi-mode and multiobjective assignment and scheduling problem. In particular, it can be modelled as a Multi-Objective and multi-mode Resource Assignment and Scheduling Problem (MOMRASP).

1.7.1 Definition

The MOMRASP can be stated as an allocation of resources to accomplish a set of tasks under an imposed set of constraints at the aim of reaching a set of predefined objectives. In its most general form, the MOMRASP can be defined as:

- **A set of tasks:**
  Each task t_i characterized by a time window representing the earliest starting and latest ending time, a priority order, and should be started if its predecessors are accomplished.
  Each task is assigned to one or more resources and can be performed according to several modes, where each mode is characterized by a specific time duration and resource consumption. Thus, any task may be executed in more than one manner depending on the availability and ability of resources to achieve such task.

- **A set of renewable resources:**
  Are available for every single period with its full capacity. In general, the resources are expensive and with high demand to achieve a set of planned tasks. Each resource has specific characteristics such as : energy power, speed, probability to succeed the accomplishment of given mission and an in-use cost. Helicopters, airborne, UAVs, Aurora aircraft (CP-140) and ships can serve as examples of the surveillance resources/assets.

1.7.2 Environment description

The resource assignment/allocation and scheduling problem to the coastal surveillance problem has been received an increasing interest by many governments in order to improve coastal security and safety, and avoid threats and illegal activities by keeping track of all activities and identifying all classes of vessels [81]. We can consider two situations for the large volume surveillance problem:

- Cooperative target search as a drowning vessel need to be detected and rescued.

- Non-cooperative target search, however, for the non-cooperative situations, the vessel attempts to avoid detection by the groups of resources (radar, UVAs, maritime platforms, satellites, etc), such as, smugglers of goods and human cargo.

Thus, the large volume surveillance problem is a challenging decision problem which, requires to find an efficient strategy for the employment of the set of surveillance assets and their networking capabilities (figure 1.1).
In what follows, we present the main existing compoments and concepts that characterize our environment.
We assume that the resources under consideration are geographically distributed. Depending on its capability and ability, each resource is able to detect, identify and track a set of targets. Each surveillance mission triggered can be subdivided into a set of elementary tasks and receive priority order to express it relative urgency.

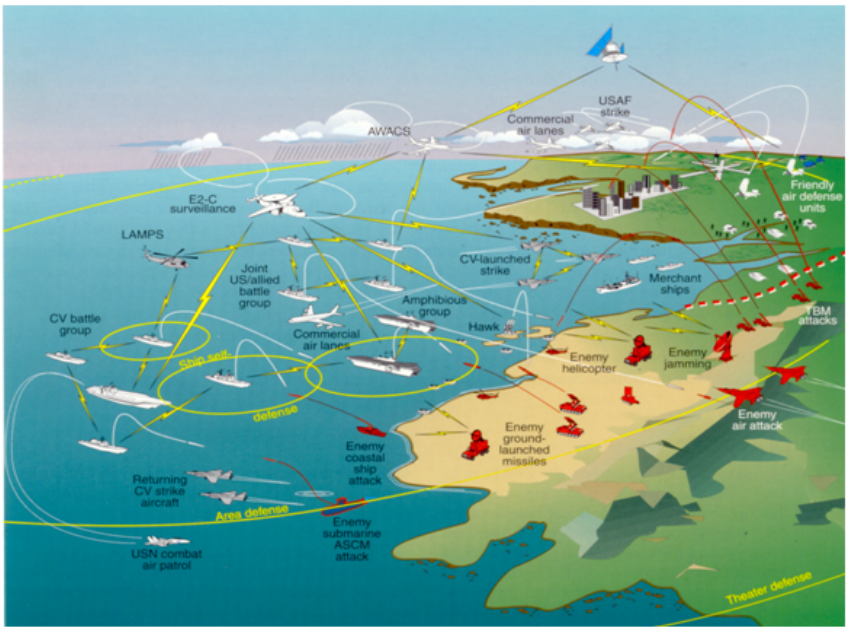

Figure 1.1: Large scale surveillance problem

1.7.3 The model

More formally the scheduling problem can be defined by:
A set of k resources $R = \{1,\ldots,k\}$ with limited availabilty and a set of N tasks with priority order and precedence relationship, the problem consists to allocate resources to tasks in order to achieve the overall objectives.

- **Objectives** The large scale surveillance problems are subject to multiple, often conflicting, objectives that should be simultaneously optimized. The objectives define the optimality of a schedule. As a result, the performance of any proposed approach to solve the MOMRASP is assessed based on their solutions quality beside the computational execution time. Thus, the evaluation of the solutions quality is according to the optimization of the objective functions.
  The problem should be designed to optimize this typical objectives:

  ❶ **Minimize makespan**: is the most applied objective in project scheduling problems. It is defined as the total time span between the start and end of the project.

  ❷ **Minimize total cost**: is the minimization of the costs of resource consumption.

  ❸ **Maximize probability of success**: The maximization of the probability of success has an important role for the surveillance problem.

- **Constraints** The MOMRASP includes various forms of constraints differing in difficulty and complexity which makes the problem extremely hard to solve. The constraints must be satified because they define the feasibility of a schedule. The constraints of the problem can be classified as:
  Task constraints

  - Precedence relationship
  - Priority of tasks

  Resources constraints

- Availabily of resources.
- Resources abilities
- Resources autonomy

1.8 Past researches on ASPs

Over the last decades, considerable efforts were dedicated to solve the assignment and scheduling problem (ASP), giving rise to a large variety of models and formulations. This interest was motivated by many potential applications like software development, wireless systems and surveillance problems Grob [36]. Important characteristics and variants of the ASP were distinguished: single mode *vs* multi-mode, mono-objective *vs* multi-objective, with preemptive *vs* non-preemptive tasks, renewable vs non renewable resources, with regular *vs* non-regular objective, and deterministic *vs* dynamic model, etc. Such variants have been investigated by Yang et al. [108]. The related literature can be viewed according to two main classes of solution approaches: exact and heuristic methods. Exact methods are the mathematical optimization techniques that consist to model the problem as mixed-integer linear programming (MILP) or mixed-integer nonlinear programming (MINLP). The MILP are often solved by Branch and Bound method and also by using commercial solvers as CPlex and Lindo. The exact methods are often used for optimizing small-sized problem with a single objective. The heuristics methods form a class of powerful solution techniques for tackling large-scale ASPs.
This section is an overview of the solution methods that have been addressed in literature to overcome the ASP. The first section deals with exact and heuristic solution procedures for ASP. Next section, presents the bi-level programming problem and provide some related works.

1.8.1 Exact methods

A quite number of different methods for optimally solving the ASP are proposed in the literature, such as linear programming methods, branch & bound procedures and

dynamic programming.

Linear programming based techniques

Linear programming is one of the most successful disciplines within the field of operations research. The linear programming model has been applied in a large number of areas including military applications, transportation, scheduling, inventory management, telecommunication, agriculture and more. The classical tool for solving the linear programming problem in practice is the class of simplex algorithms proposed and developed by George Dantzig [18].
To our knowledge, the first formulation has been presented by Pritsker et al. [85] in 1969. They proposed a discete time linear formultation for solving the scheduling problem under resource constraints such that the makespan is minimized. Slowinski [164] proposed the first exact method for the multi-mode assignment problem. For solving the problem under resource availability constraints, the author proposed a two-stage linear programming approach. Mingozzi et al. [79] proposed a linear programming formulation based on the notion of feasible subsets. A subset of activities between which no precedence relation is specified and that, if scheduled in parallel, do not violate the resource constraints. However, too many variables generated are not conducive to the optimum resolution. Liu et al. [159] studied the maximal lifetime scheduling problem for k to 1 sensor-target surveillance system. They proposed an optimal solution method to schedule sensors to observe target such that the lifetime of the system is maximized. The problem was addressed in three steps: first, they computed the maximal lifetime of the system based on linear programming method; second, they decomposed a workload matrix into a sequence of schedule matrices by using perfect k-matching method and finally, they obtained target watching timetable for the surveillance sensors.

Branch & Bound procedures (B&B)

B&B is the most widely used solution technique for solving project scheduling problems. As already mentioned in the first chapter, the resource constrained project

scheduling problem belongs to the set of complex combinatorial problems and very often B&B is the only technique which allows for the generation of optimal solutions within an acceptable computational effort. In the description of the B&B methodology we shall basically follow the paper by Agin [1].
Speranza and Vercellis [96] proposed a hierarchical two stage decomposition process for the nonpreemptive multi-project planning and scheduling problem. They developed a branch and bound algorithm. Hartmann and Sprecher [39] have shown that this algorithm may not be able to find optimal schedule with two or more renewable resources and can not enumerate all the schedules and generally suboptimal solution can be found. Demeulemeester and Herroelen [22] proposed a depth-first branch-and-bound algorithm to solve the single mode project scheduling problem under precedence and resource constraints where the objective consists in minimizing the makespan. Later, Sprecher et al. [97] extended the proposed work of Demeulemeester and Herroelen by introducing mode alternatives. More recently, Heilmann [40] proposed a branch and bound algorithm to solve the multi-mode resource-constrained project scheduling problem with minimum and maximum time lags such that the makespan was minimized. Zhu et al. [112] proposed a branch and cut algorithm. Ranjbar et al. [86] proposed a depth-first baranch and bound procedure based on arc insertion for minimizing total weighted resource tardiness penalty costs in the resource-constrained project scheduling problem. To prevent infeasible and dominated solutions, Ranjbar et al. [86] developped seven bounding rules based on the problem characteristics.

Dynamic programming (DP)

Dynamic programming was introduced by Richard Bellman during 1940s. It is a sequential optimization method used to optimize a given objective function called reward or cost function where decision are discrete. The DP is considered as an efficient procedure to determine the optimal solution to a sequential decision problem where this latter satisfy some properties such as:

- The problem must have overlapping substructure : it mean that the problem can be broken down into subproblems or stages and the objective to select the optimal decision at each stage.
- the problem must have optimal substructure : the optimal solution of local problem can lead to the optimal solution of the entire problem, means that the problem satisfy the optimality principle.

Robinson [88] provides a conceptual dynamic programming framework for solving the MRCPSP in general project networks, but does not provide any algorithm for its implementation. Hindelang and Muth [45] provide a complete dynamic programming formulation for use in decision CPM networks. Later, De et al. [20] have shown that the Hindelang and Muth approach [45] is flawed because it allows two immediate predecessors of a merge node to assume different start times at that node. Recently, Ng and Sancho [80] modeled the surveillance of a sequence of regions to be searched by one maritime patrol aircraft as mono-objective travelling salseman problem with the minimization of the mission time as an optimization objective function. A dynamic programming formulation was proposed for handling the regional surveillance problem.

These exact optimization methods are considered powerful techniques to find optimal solutions for sequential decision problems by finding the suitable decision for all possible states at each stage. However, for large-scaled and multi-objective decision problems, none of these procedures can be used since the solution space is too large and they are not able to generate an optimal solution in reasonable computational time. Therefore, it is necessary to adopt better methodologies to search the efficient solutions space that satisfy the decision maker in reasonable time. Heuristic methods appear to be powerful solution techniques for handling large-sized decision problems.

1.8.2 Approximation approaches

Full space optimization methods

ASP is $\mathcal{NP}$-hard, many heuristics were proposed to solve the problem in reasonable computation time. The related approximation approaches can be classified on three main categories: Dispatching rules, Artificial Intelligence(AI)-based approaches and Agent-based approaches. Dammak et al. [19] studied the problem of assigning a set of independent exams to a set of classrooms with limited capacity. They formulated it as 0-1 linear integer program. The problem consists only in satisfing two sets of constraints: each exam should be assigned to a classroom having total capacity greater than the number of students, and no more than one exam in each classroom. They used a simple heuristic procedure called max-size assignment to solve the exam-classroom assignment problem. The proposed approach assigns exams to classrooms in such a way that the two constraints are satisfied. If no feasible assignment was found they relax the second constraint of the problem and allow to have two exams in the same classroom. However, the heuristic can fail to find a feasible assignment even with the relaxation of the second constraint. Lee and Lee [64] proposed a hybrid search approach based on genetic algorithm and ant colony optimization to solve the mono-objective resource allocation problem. The genetic algorithm was used to provide the diversity in search space and ant colony was performed to exploit best solution. The addressed problem formulation is based on two assumptions: all the resources must be assigned to activities; and an individual allocating probability P_{ij} for the fulfill of the ith activity by assigning the jth resource is known for all i and j. They tried to minimize the penalty of the ith activity not being fulfilled by its assigned resource. Belfares et al. [7] proposed a progressive resource allocation method based on the tabu search heuristic and a posteriori approach to solve the multi-objective RCPS problem with multiple modes. They used a Multi-criteria Filtering Procedure and the Dominance Rule to filter and to choose the efficient solutions. For the assignment problem, we refer the reader to Chaharsooghi and Kermani [15], Dai et al. [17], Korhonen et Syrjänen [60] and Lin and Gen [67]. Mendes et al. [162] proposed a random key-based genetic algorithm for the single mode resource constrained

project scheduling problem by adopting an indirect representation of the chromosome solution. The generated chromosomes do not represent the solutions to the original problem and special concepts can be used to derive the solutions. They classified the schedules into three types. First type is the semi-active schedules: where feasible schedules are obtained by sequencing activities as early as possible. No activity can be started earlier without changing the sequence. Second, the active schedules: in which no activity can be delayed without delaying some other activity or breaking a precedence relationship. Optimal schedules are always members of this set of schedules and active schedules are members of the set of semi-active schedules. Third, the non-delay schedules: in which no resource is allowed to be idle when it could start to process an activity. Non-delay schedules are also members of the set of active schedules. In their work, Mendes et al. [162] limited the search space in the set of active schedules. However, the active schedules space is very large and beside containing the optimal schedule, it usually contains large number of solutions with poor quality in terms of makespan. They used, the concept of parameterised schedules to reduce this search space by placing a restriction on the project duration and to generate the direct solution. They employed a unique fitness measurement criterion termed modified makespan to differentiate between schedules having the same makespan. Alcaraz and Maroto [3] proposed a self-adaptation of the genetic algorithm to solve the single mode resource constrained project scheduling problem in such way the makespan was minimized. The initial population was not generated randomly but they used a sampling method based on the Latest Finish Time as the selection priority rule. They added an additional gene into the solution chromosome in order to control the way in which the phenotype was built. This gene called forward/backward gene, which determines whether forward or backward pass scheduling has been employed. Loukil et al. [161] proposed a Multi-objective simulated annealing method to solve the multiple objectives production scheduling problem. They presented the model as sequences of permutation of jobs in each machine and they used three way to define the neighbor of a sequence: first, two jobs are exchanged; second, a job is inserted and the third, the neigbhor was chosen randomly. They used a weighting function for the determination of a good direction on the efficient frontier. Trojet et al. [102] proposed

a decision support framework for industrial application that consists in building the production planning under resource constraints modelled as a constraint satisfaction problem. They developped an approach based on the concept of cumulative global constraint in order to find an optimal solution that minimizes the project makespan. However, if no generated solution respects the delivery deadline, they can consider some constraint relaxations.
Kolish and Hartmann [58],[59], Brucker et al. [14] and Herroelen et al. [42] published different surveys of the heuristic approaches for solving the scheduling problem.

Decomposition methods: Bi-level optimization approaches

The bilevel programming model is a special case of multiple level programming problem. There are two decision makers that have different objective functions, variables and constraints and they are working toward satisfying their own objectives. The decision process is sequential, the upper level decision maker (the leader) makes his/her decisions first, anticipating those of the lower level decision maker (the follower). While the follower decision is affected by the leader choices, the decision of the leader is also affected by his/her followers optimal reactions. Therefore, to identify a solution for the leader who has multiple and possibly conflicting objectives, there is a need to consider both the solution of the leader and his follower's decision.
The bi-level optimization problem can be stated as:

P1 $min_{x \in X, y}\ F(x, y)$

s.t $G(x, y) \leq 0$

P2 $min_{y}\ f(x, y)$

s.t $g(x, y) \leq 0$

The variables of problem are divided into two classes, the upper-level variables $x \in$

R^{n1} and the lower-level variables $y \in R^{n2}$. $F : R^{n1} \times R^{n2} \in R$ and $f : R^{n1} \times R^{n2} \in R$ are the upper-level and lower-level objective functions respectively, while the vector-valued functions $G : R^{n1} \times R^{n2} \in R^{n1}$ and $g : R^{n1} \times R^{n2} \in R^{n2}$ are called the upper-level and lower-level constraints respectively. Upper-level constraints involve variables from both levels and play a very specific role.

Bilevel programming model is usually complicated and hard to resolve because the solution of the lower level affects the solution of the upper level. The existing methods for solving it can be divided into the following categories: methods based on vertex enumeration, methods based on KuhnTucker conditions, fuzzy approach and methods based on metaheuristics. In literature, the bi-level programming method was used for solving some scheduling and assignment optimization problems. Guo et al.[38], proposed to solve the multi and mixed model apparel assembly line as a multi-objective scheduling problem such that the total earliness/tardiness penalty cost is minimized and the production flow is maximized without preemption. They developed a bi-level GA to handle the problem of two orders. The first level conceived to generate a feasible assignment of operations to workstations. In the second level, the algorithm determines the starting time of each scheduling status in each cycle. Kreucher et al. [62] studied the sensor scheduling problem, which consists in detecting and tracking smart targets. Smart targets means target that can detect and react when they are under surveillence. They used the reinforcement learning approach to solve the scheduling problem in two steps. During the first step, they used a detection algorithm to decide on the presence or absence of smart targets. The second step, the tracking algorithm geolocate and track the target in the surveillance region. Ho and Leung [46] studied the flight loading operations for airline catering as a manpower scheduling problem with time windows and job-skills constraints. The problem consists to form teams of workers with appropriate skills and then assign teams and start time for the jobs or flights such that the number of unassigned flights are minimized. They proposed a two stage solution approach to solve this problem. In the first stage, they solve a pairing problem to form a team based on a greedy heuristic to match workers and in the second stage, jobs are assigned to teams based on a tabu search heuristic. However, a bad team formulation in the first stage can

lead to difficulties in generating feasible solutions for the second stage. Horn et *al.* [49] developed an optimization procedures to schedule the activities of patrol boat and their crews based on simulated annealing and specilized heuritic techniques. They adopted a multi-stage scheduling scheme for handling this scheduling and staff rostering problem with multi-criteria feature. However, they consider the traveling time of the patrol boat to reach or return from a deployement mission to port as negligible. Lasserre [63] proposed a top-down hierarchical approach to solve the job shop planning and scheduling problem. He used a decomposition approach that partitions the entire problem into two subproblems and alternate between them: first, solving a planning problem for a fixed choice of sequence of products on the machines with minimum cost function that include the holding cost, backlogging cost, production cost and set up cost. Then, finding a schedule compatible with the fixed production plan with minimum makespan as criterion and due date as constraints. Erdirik-Dogan and Grossmann [30] studied a multi-period mixed integer linear programming model for the simultaneous planning and scheduling of single-stage multi-product continuous plants with parallel units, which is an extension of the single unit model proposed in [29], based on continuous time representation. To solve this problem for long time horizons, they proposed a bi-level decomposition algorithm in which the original problem is decomposed into an upper level planning and a lower level scheduling problem. In the upper level, They determined the optimal sequence of the production during each time period with maximum profit as criterion. In the lower level the original scheduling model is solved for a subset of products as determined by the upper level. However, the upper level model is the bottleneck in computational time due to the incorporation of the sequencing decisions. Lin et *al.* [68] developed a bi-level intelligent scheduling approach to schedule jobs on capacitated multiple flowline production systems. They integrated the lot sizing and sequencing decisions in order to improve the system. The product-level scheduling is first performed to generate the daily production plan, including production type and quantity on each day over a week based on the utility value. Board-level scheduling is then used to determine the production schedule to reduce tardiness and capacity and improve the line's productivity. However, the method can introduce more work in process inventory. Karlof and Wang

[54] proposed a bi-level formulation of the flow-shop scheduling problem. The shop owner is regarded as the upper level, and consists in assigning the operations to the machines in order to minimize the flowtime, while the customer is represented as the lower level, and consists in deciding the job's ordering to minimize the makespan. They developed a bi-level branch and bound algorithm to find the global optimal solution. However, the algorithm able to solve only small size problems (until 8 jobs and 8 machines) with a considerable computational time (more than 12H). Zheng et al. [111] developed an interactive method based on satisfactoriness as well as multi-objective optimization at the upper level and a measurement function at the lower level for solving the bi-level multi-objective programming problem. They used the ε-constraint method to obtain a set of efficient solutions that satisfy the upper level decision maker. Then, at the lower level, they examined the generated solutions using a measurement function until a best satisfactory solution is reached. However, the final solution is always efficient to the upper level problem which is considered in the dominant position and they suppose that the decision makers between the two levels are cooperative.

Many attempts have been made to solve the bi-level programming problems in the past decades, unfortunately most of them are either incapable of finding global optima or computationally very intensive for problems of realistic size.

1.9 Conclusion

We defined the project scheduling problems under their different classes and variants. In this context, the resource-constrained project scheduling represents the most studied problem in project scheduling. As a matter of fact, several methods have been proposed for solving it.

The following chapter is reserved to the literature review on resource assignment and task scheduling problems.

Chapter 2

Population Based Optimization Approaches

2.1 Introduction

A large number of large scale real-life optimization problems encountered in science, industry, engineering, economics, management, defense and business are often complex and hard to solve. Thereby, with the continuing increase in computational power, both researchers and practitioners are concerned to boost up and enhance methods and models in attempt to reach the predefined objectives and fulfill the harsh constraints. Exact methods are not entirely appropriate within a reasonable computational time. Motivated by this shortfall, the last three decades have witnessed the development of numerous metaheuristics approaches. These approaches are well adopted to find efficient solutions for extended sized problems within an affordable computational time. In its first signification, a metaheuristic represented an exlporation method ables to search very large spaces of candidate solutions. However, metaheuristics do not guarantee an optimal solution is ever found. Metaheuristics serve three main purposes: solving problems faster, solving large problems, and obtaining robust algorithms. Moreover, they are simple to design and implement, and are very flexible.

This chapter provides the terminology and the key concepts of the different proposed methods to tackle hard optimization problems.
The chapter is structured as follows: Section 2.2 .. Optimizing involves finding parameter values for the model that minimize the misfit function.

2.2 Evolutionary algorithms

The Evolutionary algorithms (EAs) are a global search techniques inspired by the biological evolution of the species based on the Mendel's rules and the fundamental principle of Charles Darwin. In the 1980s, these theories of creation of new species have inspired computer scientists in designing evolutionary algorithms. This evolution is guided by the fitness evaluation or the individuals' performance in the population. Indeed, EAs was considered as a class of iterative optimization algorithms that simulate the evolution of species via the selection process, mutation and reproduction. They are based on the evolution of a population of individuals. Contrarily to the traditional optimization methods, which are considered as a sequential search method that start from an initial feasible solution and try to improve it by moving to their neighbor. EAs use a popoulation of potential solutions which evolves from a generation to the subsequent through the selection rules and operators.
EAs are classified as a stochastic population-based metaheuristics that have been successfully applied to a wide variety of real and complex optimization problems to overcome the drawbacks of exact optimal methods (multimodal, multiobjective, and highly constrained problems). They are the most studied population-based algorithms. Their success in solving difficult optimization problems in various domains such as combinatorial optimization, planning and control, engineering design, data mining and machine learning has promoted the field known as evolutionary computation.
During the past 40 years many efforts were undertaken to understand the natural evolution processes and to model them algorithmically leading to the emergence of three independently main models of evolutionary algorithms:

- Genetic algorithm (GA), mainly developed by Holland [48]. It is based on the mechanisms of heredity and surviving rule of the natural biological evolution;
- Evolution strategies (ES), developed by Rechenberg [87]. They are designed for continuous optimization problems using continuously varying parameters;
- Evolutionary programming (EP) by L. Fogel [31]. Later, in the end of 1980s, GP has been proposed by Koza [61]. The EP uses only mutation and selection operators.

Each of these constitutes a different approach which are now collectively denominated as EAs and have been proven for single objective optimization problems. Nevertheless, GA represents the very popular subclass of EAs.
For solving optimization problems, Algorithm 2.1 summarized the evolutionary algorithm principale.

Algorithm 2.1: EA template

INPUT
P_0: initial population
INITIALIZATION
Fitness evaluation of P_0
$i \leftarrow 1$
$P_i \leftarrow P_0$
LOOPING PROCEDURE
DO
Select parents (P_i')
Recombine parents (P_i')
Mutate (P_i')
Evaluate_fitness (P_i')
$R_i \leftarrow$ Replace (P_i',P_i)
$i \leftarrow i + 1$
WHILE *stopping criteria not met*
OUTPUT
P_{i+1}: The updated population at generation $i + 1$

2.2.1 Genetic algorithm

The GAs owe thier success and popularity due to the simplicity and efficiency when handling for more complex optimization problem. The origins of their use for solving problems go up to the 60s, when John Holland in collaboration with his group of researchers at the university of Michigan took into consideration the crossover operator in complement with mutations. They discovered that this operator is the one that allows, in most times, to reach the optimum of a function by combining the different individuals' genes.

The GA principle can be simply described with a case of a single objective minimization problem. The method strats with an initial set of n solutions, which can be

generated randomly, constitutes the initial population to be improved during a succession of iterations called generations until a termination criterion by maintaining its size constant. Each individual x of the population has a fitness value $f(x)$ that determines its probability of survival to the subsequent generation according to the aimed objective. The smaller $f(x)$, the higher the probability of the x to be preserved in the next generation. The improvement of the population at each generation is obtained by the genetic operators: the selection operator that determines the retaining individuals to survive and to reproduce in the next generation, and the reproduction operator that insures the recombination of the genetic material of parental genes to create offspring with new potentialies.

In pratice, each member of the population is coded as an array or a string of specific length forming an unnaturel chromosome. Variation into population are engendered by the recombination operators: crossover and mutation.

Crossover operation consists to combine the genes chains of two chromosomes in order to get new chromosomes. The crossover can be simple or multiple. The first consists in exchanging portions of the two chains in a single point while the second includes several points of crossing. Mutation is a random operation that consists in substituting a gene to another one within a chromosome in order to diversify the solutions and produce new points in the solution space (figure 2.1). As the ASP belongs to the class of strongly $\mathcal{NP}$-hard combinatorial optimization problems (Blazewiez et al., 1983), a large variants and extensions have been proposed on literature (Hartmann & Briskorn 2010). Numerous solution methods based on evolutionary algorithms have been suggested to solve a large variants and extensions of the assignment and scheduling problem (ASP). Mori and Tseng [129] conceive a single objective genetic algorithm for scheduling problem with multiple mode. Alcaraz and Maroto [113] propose a self-adaptation of the genetic algorithm to solve the single mode resource constrained project scheduling problem in such a way the makespan is minimized; whereas Hartmann [125] proposes a genetic algorithm for solving the multi-mode resource-constrained project scheduling problem (MM-RCPSP) with makespan minimization. He used an activity list (AL) and a mode assignment list for chromosome representation. Later, Alcaraz et al. [114] apply a similar framework, trying to

overcome Hartmanns GA drawbacks. Likewise, Lova et al. [130] proposed a new normalized fitness function in their hybrid genetic algorithm for solving MM-RCPSP to overcome some drawbacks in Alcaraz et al. [114]. Lee and Lee [158] propose a hybrid search approach based on genetic algorithm and ant colony optimization to solve the mono-objective resource allocation problem. The genetic algorithm was used to provide the diversity in search space and ant colony was performed to exploit best solution. Damak et al. [121] solve MM-RCPSP with a differential evolution algorithm. Mendes et al. [162] present a random key genetic algorithm for the single mode scheduling problem with makespan minimization.

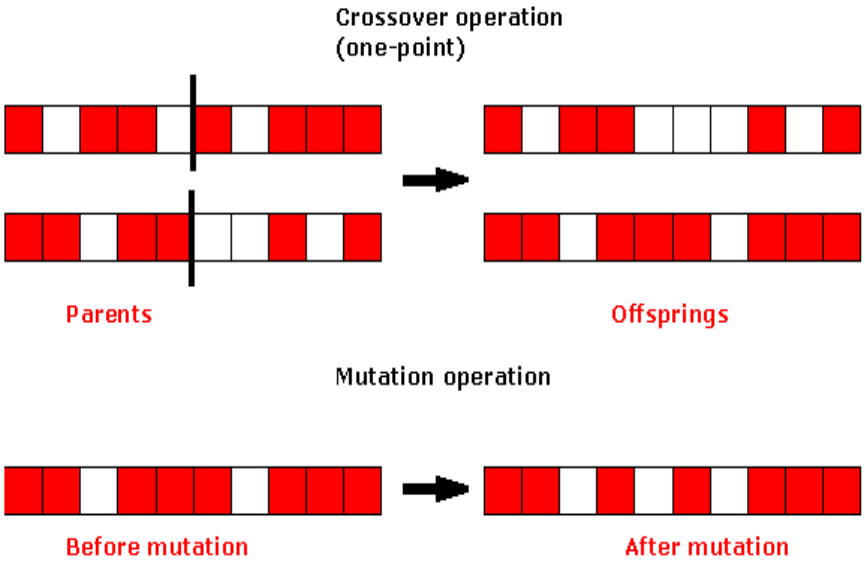

Figure 2.1: Crossover and mutation operators

2.2.2 Multi-objective evolutionary algorithms

Most real world optimization problems encountered in practice often involve multiple objectives that should be simultaneously optimized with respect to a set of constraints. These objectives are often conflicting and incommensurable. Improvement of one objective may lead to the deterioration of another. Finding a single

optimal solution that optimize all the objective functions simultaneously is not possible. Instead, a set of efficient solutions presenting the optimal trade-offs between the different objectives is to be generated, called the Pareto optimal solutions. When plotted in the objective space, this set of compromise solutions is called the Pareto front. The Pareto optimality is a concept used in multi-objective optimization and was first proposed by Edgeworth and Pareto [],

Definition 1 Pareto dominance. *An objective vector* $u = (u_1, \ldots, u_m)$ *is said to dominate another objective vector* $v = (v_1, \ldots, v_m)$*, denoted as* $u \prec v$*, iff* $\forall i \in \{1, \ldots, m\}$*:* $u_i \leq v_i \bigwedge \exists i \in \{1, \ldots, m\}$*:* $u_i < v_i$ *and* $u \neq v$*.*

Definition 2 Pareto optimality. *A solution* $x^* \in S$ *is Pareto optimal if for every* $x \in S$*,* $F(x)$ *does not dominate* $F(x^*)$*, that is,* $F(x) \not\prec F(x^*)$*.*

Definition 3 Pareto optimal set. *For a given MOP* (F, S)*, the Pareto optimal set is defined as* $P^* = \{x \in S / \nexists x' \in S, F(x') \prec F(x)\}$*.*

As a sequel, the complexity of handling a Multi-objective Optimization Problems (MOPs) is to find a well-converged and well-distributed approximation of the Pareto front. Thereby, due to their population-based nature, EAs have received an increasing interest given their efficiency to produce a good approximation set in a single run when solving MOPs. Since the Schaffer's work [131], these EAs are called multi-objective EAs (MOEAs). Several approaches based on multi-objective EAs (MOEAs) are proposed in literature to improve the generation of good solutions. Some of them are non-Pareto approaches such as vector evaluated genetic algorithm (VEGA) proposed by Schaffer [131], weight based genetic algorithm (WBGA) and the lexicographic genetic algorithm of Fourman [119]. Unfortunately, these methods present the same drawbacks as the weighted aggregation techniques such as the sensibility to concavity of the Pareto front and the inability to find Pareto optimal solutions. Other approaches, are based on the concepts of Pareto optimality in their search procedures to ensure two main goals: the convergence towrads the Pareto frontier and the diversification of the solutions spread all over the Pareto frontier. Among the pioneering

algorithms are the multi-objective genetic algoritm (MOGA) proposed by Fonseca and Fleming [118] and the non-dominated sorting genetic algorithm (NSGA) conceived by Srinivas and Deb [132]. An enhancement of these algorithms by the elitist methods is proposed by Zitzler and Thiele [134] in their strength Pareto evolutionary algorithm (SPEA).

Other approaches, are based on the concepts of Pareto optimality in their search procedures and can be classified as non-elitist and elitist methods. Elitist guarantees that elite solutions are preserved in the next population such as strength Pareto evolutionary algorithm (SPEA) proposed by Zitzler and Thiele [134], non-dominated sorting genetic algorithm II (NSGA II) as well as indicator based evolutionary algorithm (IBEA). The non-elitist methods due to its applicabilty in real-world situations.

2.3 The ant colony optimization algorithms

The ant colony optimization algorithm (ACO), is a stochastic search procedure that belongs to the swarm intelligence class. Which is inspired by the behavior of real ant colonies while they searching for food source. At the core of this behavior is the indirect communication between the ants by means of chemical pheromone trails, which enables them to find short paths between their nest and food sources. This characteristic of real ant colonies is exploited in ACO algorithms in order to solve difficult optimization problems.

Initially, the ACO metaheuristic has been proposed by M. Dorigo in [24] for solving the well-known traveling salesman problem (TSP). The main motivation for using ACO metaheuristic by researchers is the adaptable character of the latter which makes of it a very useful tool to tackle a wide variety of combinatorial optimization problems, as well as the simplicity of the model on which the method is based. This metaheuristic is especially suited to find solutions for difficult optimization problems. The ACO metaheuristic is composed of artificial ants that cooperate to find good solution.

In ACO metaheuristic, a colony of cooperating agents searches for a good solution to an optimization problem. Each agent or ant can already find a solution or at least a

part of solution on its own but the optimal solution can not be reached individually. According to the problem considered, ants are given a starting state and they move through a sequence of neighbor states trying to find a shortest path. Moves are based on a stochastic decision policy directed by the ants'internal state, the pheromone trails and the environment information. Generally, the amount of pheromone deposited is proportional to the quality of a move an ant has made. Thus, better solution is obtained with more pheromone depositing. After an ant has found a solution, it dies. The main tenet of the ACO metaheuristic is the use of a priori information concerning the problem data combined called heuristic information with a posteriori information concerning the structure of previous solutions obtained called pheromone trail. This combination is the stochastic decision policy called transition rule. Three main criterion have to be defined to implement an ant algorithm:

- The Pheromone trail: Cooperation and indirect communication between ants is the basic tenet to find good solutions quality to difficult optimization problems, and this is through the pheromone trail. It is considered as a numeric variable where each artificial ant of the colony can read or write information concerning the problem representation.

- The transition rule: It is applied to determine the next position of the ant and it is based on two parameters: the pheromone trail which represents the posteriori information and a heuristic information that represents the priori information. In This respect, each ant of the colony apply the transition rule in order to choose the next state to move to. Thus, the transition rule is a stochastic function which influences the process of selection the next state.

- Pheromone update: The pheromone update is essential and basic phenomenon to avoid stagnation situation, i.e. in which all ants build at each cycle the same solution, and to motivate the exploitation and exploration of the search space.

Algorithm 2.2 summarized the steps of the ACO algorithm.

Algorithm 2.2: ACO steps

INPUT
f: number of ants
INITIALIZATION
Initialize the pheromone trails
Assign the ants to the starting node
LOOPING PROCEDURE
DO
Move to the next node by applying the transition rule
Update the pheromone trails
WHILE *stopping criteria not met*
OUTPUT
Best solution found or a set of solutions

2.4 Particale swarm optimization

The particale swarm optimization (PSO), introduced by Kennedy and Eberhart [57], is one of the population based metaheuristics and it belongs to the swarm intelligence class. It is a recent efficient metaheuristic which is inspired from animal grouping behaviour such as bird flocking and fish schooling to find a place with enough food. Researchers simulate the social behavior of the birds in seraching food. The birds determine their velocities by means of two factors: their best past experience and the best past experience of the group. These simulations have highlighted the ability of individuals of a group to maintain a distance between them and follow a global movement relative to their local neighborhood. Thus, the PSO mimic the bird natural behaviour when searching for food.

In PSO a swarm of particles are placed in the search space. All particles have fitness values which are determined via the objective function to optimize. Thus, a particle i represents a candidate solution in the search space and has its own position x_i and

a vector of change in position (velocity) v_i, which guide the flying direction and step of the particle. In each iteration, a particle determines its movement through the search space by combining some aspect of the history of its own current and best (best-fitness) locations with those of one or more members of the swarm, with some random perturbations. Thus, the movement of each particle can be characterized as: $x_i(t) = x_i(t-1) + v_i(t-1)$. Then, the next iteration will takes place after all particles have been moved. Eventually the swarm as a whole, like a flock of birds collectively foraging for food, is likely to move close to an optimum of the fitness function. To apply the PSO, some parameters should be defined in advance such as:

- Solution encoding: it represents the solution space via the evaluation of a D-dimensional vector.
- Interia weight: it control the impact of the precedent velocity on the current one. Thus, for large values of the inertia weight, the influence of the precedent historic velocities will be much higher. Thus, the inertia weight represents a trade-off between global exploration and local exploitation.
- Maximum velocity: it defines the amount of change that will be applied to the particle. Therefore, it is necessary to limit it by a threshold value.

Algorithm 2.3 summarized the PSO algorithm principale.

Algorithm 2.3: PSO template

INPUT
P: initial population
INITIALIZATION
Fitness evaluation of each particle in P
LOOPING PROCEDURE
DO
Update velocity
Move to the next position
Evaluate all solutions
Update particle memories (x_i,v_i)
WHILE *stopping criteria not met*
OUTPUT
Best solution found or a set of solutions

2.5 Memetic algorithms

Memetic algorithms (MA) have been considered as a recent areas of research in the field of evolutionary computation. They belong to the class of stochastic search methods in which EAs are combined with local search heuristics techniques, approximation algorithms or, even exact methods.

2.6 Conclusion

By reviewing the different combinatorial optimization methods provided in the literature, we conclude that evolutionary algorithms are increasingly gaining popularity to solve multi-objective optimization problems. Since the past two decades, several approaches have been proposed to search for a set of potential optimal solutions. Therefore, the performance of such algorithms is assessed based on the diversity of

solutions and their proximity to the pareto front.

In the next chapter, we propose a multi-criteria genetic algorithm to achieve balanced proximity-diversity of the generated solutions for multi-objective and multi-mode ASP.

Part II

Contributions

Chapter 3

A Genetic Local Search Approach for the Single-mode Resource Allocation and Scheduling Problem

3.1 Introduction

The resource-constrained project scheduling problem (RCPSP) is a well-known problem in scheduling theory that has been applied to many applications, such as manpower scheduling [150], grid computing [160], classroom arrangement [147] and surveillance problem [26][159]. It is widely known to be notoriously difficult to solve due to the employment of scarce resources as well as precedence relations between activities. Considering these constraints, the problem consists to find an efficient arrangement of activities that leads to a minimal completion time of the project.

Besides the problem's relevance with respect to the applicability in real-world situations, the RCPSP has attracted a considerable attention from both researchers and practitioners. Blazewicz et al. [141] have proved that the RCPSP belongs to the class of the strongly $\mathcal{NP}$-hard problems. Therefore, a very large literature is devoted to solve it, given place to various exact and approximation methods on the wide variety of variants and assumptions. Exact methods involve the use of enumerative schemes

and often used to solve at optimatily small-sized problems [149, 164, 165, 166, 169]. However, these algorithms are far from being efficient in terms of computational time when solving large scale problems. Approximation methods and more specifically, metaheuristic algorithms appear to be robust in finding good solutions in a reasonable computational time. The related approximation approaches include the genetic algorithm [136, 137, 138, 139, 152, 162], The particle swarm optimization [144], the simulated annealing [142, 161], the tabu search [140, 163], the ant colony optimization [146, 26, 150, 160] and hybrid algorithms [145, 148, 158, 167, 168]. We refer the reader to Kolish and Hartmann [155] [156], Brucker et al. [143] and Herroelen et al. [154] for a comprehensive review on RCPSPs.

In this chapter, we develop a genetic programming approach for solving the RCPSP. Solutions generated with standard genetic operators may tend to create infeasible individuals during the search because of the discrete and often constrained search space. Moreover, prematurely convergence may lead to search stagnation on restricted regions of the search space instead of convergence towards unexplored area. In this respect, considering the exploration ability of GAs, the exploitation capability attained by local search procedures and in order to deal with the complexity of both objective space and constraints, we propose to combine the local search engines within the genetic method. The obtained Genetic Local Search Algorithm (GLSA) tries mainly to generate potential sub-optimal solutions. The GLSA runs GA as the main algorithm and thereafter uses the LS procedure to improve the results provided by GA.

On the basis of the state-of-art algorithms, we present the results of a comprehensive computational study, carried out on 960 benchmark instances, that provides strong evidence that the proposed GLSA is highly competitive.

The remainder of this chapter is organized as follows. In section 2, we present the problem description and provide its mixed-integer linear formulation. A GLSA for the RCPSP is presented in section 3. The empirical performance of the proposed approach is assessed in section 4. Finally, we provide in section 5 the concluding remarks and directions for future research.

3.2 Model description and formulation

The RCPSP can be stated as follows. A project with n activities labeled $i = 1, \ldots, n$ has to be planned without preemption. Due to technological requirements, precedence relations are to to be pointed out. These relationship are given by sets of immediate predecessors P_i indicating that an activity i may not be started before each of its predecessors $j \in P_i$ is completed. Precedence relations can be represented by an activity-on-node network which is assumed to be acyclic. There are $|R|$ renewable resource types with r_{ik} denoting the resource requirements of activity i with respect to resource k.

The objective of the problem is to schedule each activity of a project subject to precedence and resource constraints, in order to minimize the project makespan. Based on the above definitions, the problem can be represented formally as the following time indexed formulation:

$$Min \;\; Z(X) = Max_{i=1}^{n} \, F_i \tag{3.1}$$

subject to

$$\sum_{i=1}^{n} r_{ik} \, x_{ik} \leq E_k, \; \forall k = 1, \ldots, |R| \tag{3.2}$$

$$\sum_{t=0}^{T} y_i^t \leq 1, \; i \; = 1, \ldots, n \tag{3.3}$$

$$\sum_{t \in T} t \, (y_i^t - y_j^t) \geq F_j, \; \forall j \in P_i \tag{3.4}$$

$$F_i \geq \sum_{t \in T} t y_i^t + d_i, \; i = 1, \ldots, n \tag{3.5}$$

Table 3.1: Symbols for the mathematical model (1)-(6)

Indices/sets	
i	Task index, $i \in \{1, \ldots, n\}$
k	Resource index, $k \in \{1, \ldots, \lvert R \rvert\}$
P_i	Set of immediate predeccessors of task i
Parameters	
E_k	Constant amount of renewable resource k is available at any time
$[a_i, b_i]$	Time window on the completion of task i
T	An upper bound on the makespan
d_i	processing time of activity i
Decision variables	
F_i	Finishing time of task i
y_i^t	Equal to 1 if task i starts at the beginning of period $t \in T$ and 0 otherwise
x_{ik}	Equal to 1 if activity i is processed by resource k and 0 otherwise

$$x_{ik}, y_i^t \in \{0, 1\}, \forall i = 1, \ldots, n, \forall t \in T, a_i \leq F_i \leq b_i \tag{3.6}$$

- Objective function:
  Equation (3.1) symbolizes the project makespan to be minimized with respect to the problem constraints.

- Problem constraints:
  The RCPSP includes various forms of constraints differing in difficulty and complexity which makes the problem extremely hard to solve. The problem constraints include resources constraints and task constraints that should be satisfied to insure feasibility.

- Constraints (3.2) take care of the limited quantity of the renewable resource constraints.
- Constraints (3.3) impose that each task is started at most at one time over the planning horizon T.
- Constraints (3.4) ensure that the starting time of task i that follows task j is greater or equal than the completion of task j, F_j.
- Constraints (3.5) establish that the completion time of task i is greater or equal than the sum of starting times and processing times.
- Constraints (3.6) state logical binary restrictions on decision variables.

In table 3.2, a simple example with 4 activities and only one renewable resources available with 4 units in each period is illustrated. Figure **??**, shows the optimal schedule with $Z(X) = 8$, where both precedence constraint and resource constraint are satisfied.

Table 3.2: An RCPSP with $n = 4$ and $|R| = 1$

Activity i	P_i	d_i	r_1
1	-	3	1
2	1	4	2
3	1	2	2
4	$2, 3$	1	3

3.3 Model implementation

The combinatorial structure of the proposed model makes the generation of the solution difficult and time consuming. Therefore, we propose to solve it using a hybrid approach that applies local search with genetic algorithm. More generally, the aim of

this study is to improve the results provided by genetic algorithm (GA) by hybridizing it with a local search (LS) technique during the mutation process. As such, the proposed model is implemented in two main steps.

We choose to introduce the LS procedure within the GA generational loop, after the application of selection and crossover operators. Next, some individuals from the current generation are then improved by the LS algorithm in order to help the algorithm to find the global optimum and avoid an eventual prematurely convergence caused by selection and crossover operators. A solution s to the problem can be reduced to an activities list π, which it represents the activities sequence $\pi(i) = (\pi_1^t, \ldots, \pi_n^t)$, where activity i is scheduled during the period t^{th}, with $t \in T$. We call an activity list precedence feasible (PF) if every activity is positioned after all of its predecessors in the list, and resource feasible (RF) if the use of each resource $k \in |R|$ does not exceed its available amount. The local search operates as follows: we select randomly a solution $s \in S$ to be improved. The transition from one alternative s to s' is referred to as a move. A neighborhood operator is performed to shift an activity i from its current period t to another period t', $t \neq t'$, such that the precedence feasible is respected. As, we need to determine the number of solutions to be improved by LS, we define by P_{LS} the proportion of the current population that represents the individuals on which LS is applied. Additionally, running the LS procedure at each generation is too expensive. Therefore, the probability of calling the LS procedure is inversely propotional to the population size.

GLSA for RCPSP

INPUT

P_0: The initial population of size S

$cross_p$: The crossover probability

P_{LS}: Proportion of S to be enhanced

C_i: The offspring population at generation i

P_i: The parent population at generation i

OUTPUT

P_{i+1}: The updated parent population at generation $i+1$

C_i: The updated offspring population at generation $i+1$

INITIALIZATION

Randomly generate P_0

Fitness evaluation of P_0

$i \leftarrow 1$

$P_i \leftarrow P_0$

LOOPING PROCEDURE

DO

Apply the crossover operator according to $cross_p$:

Generate the offspring population C_i

Apply LS procedure with probability P_{LS}:

Generate a neighbor $N(s) = LS(s, t, t')$

If $Z(N(s)) < Z(s)$ then $s \leftarrow N(s)$

$R_i \leftarrow$ non-domination sort $(C_i \cup P_i)$

$i \leftarrow i + 1$

WHILE *stopping criteria not met*

In GA, each chromosome represents the activities-sequence alternative. The sequences are encoded as a random vector of n genes, where n is the number of activities as follows:

SOLUTION ENCODING

1	...	n
π_1	...	π_n

When applying GLSA, we need to define an evaluation function that determines the probability of survival of an individual to the subsequent generation. Usually, the fitness function coincides with the objective function as it is an accurate measure of how good the solution is. However, some individuals of good makespan can be infeasible due to the violation of PF or/and RF constraints. Therefore, their fitness should be penalized by adding the upper bound on project's makespan T computed as the summation of the maximal durations of the activities. Consequently the penalized individual will be eliminated from the genetic process.
The Elitist method [153] was used for the selection of the individuals that will survive in subsequent generation. The elitist method guarantees that the best feasible solutions of the current population are preserved in the next population. Two chromosomes are selected randomly to act as parents. In the literature random keys technique was found to be the fastest one [139], it will be adopted in our GA during the crossover phase.

3.4 Experimental analysis

In this section, we present the experimental results and analyze the computational performance of the proposed GLSA. The platform for conducting the experiments is a Core 2 Duo 1.66 GHz laptop with 2 GB RAM. The program was implemented using the development environment C++ 6.0. We examine the performance of our proposed algorithm by means of computational experiments using benchmark problems.

3.4.1 Parameter settings

Meta-heuristic algorithms require an initial decision about the values of numerous parameters. The parameter settings may impact the solution quality and speed. We have performed numerous experiments to determine an adequate set of parameter values for the GA and LS as depicted in table 3.3.

Table 3.3: Meta-tuning of the glsa

Parameters	*Values*
Population size (S)	n
Crossover rate ($cross_p$)	0.7
Generations (G)	40
Number of iterations (it)	1000
Tabu list$_\pi$ length (TL_π)	5
Proportion of population (P_{LS})	0.2

3.4.2 Performance evaluation

A series of experiments were carried out to examine the performance of the proposed GLSA presented in section three using benchmark problems from PSPLIB (Project Scheduling Problem Library http://129.187.106.231/psplib/) [157] $J30$ and $J60$ with non-dummy activities. In PSPLIB benchmark, the number of RCPSP instances for $J30$ and $J60$ is 480 respectively. We choose the maximum number of iterations as the termination criteria set to 1000 iterations. The proposed algorithm is compared with other approaches proposed in literature. For the data sets $J30$ the optimal makespans for all instances are known. However, for $J60$ the optimal makespans for instances are not known and only lower bounds are provided.

As, The optimal makespans (C_i^*) for all instances in $J30$ are known, the quality of

generated schedule s can be obtained via the deviation from the optimal solution Dev_i of instance i as follows:

$$Dev_i = \frac{s - C_i^*}{C_i^*} \tag{3.7}$$

Moreover, Dev_{avg} is the average deviation from the optimal makespan for ProGen $J = 30$:

$$Dev_{avg} = \frac{\sum_i (Dev_i)}{|instances|} \tag{3.8}$$

We have compared the GLSA with 14 existing algorithms in the literature based on

Table 3.4: Average deviations from the optimal solutions of $J30$

Algorithm	*Reference*	*Year*	Dev_{avg}
Random Key GA	Mendes et al. [162]	2009	0.06
Neurogenetic	Agrawal et al. [136]	2011	0.13
ACOSS	Chen et al. [145]	2010	0.14
GLSA	**This study**	2013	**0.19**
ANGEL	Tseng and Chen [167]	2006	0.22
GA	Alcaraz et al. [138]	2004	0.25
Scatter Search	Debels et al. [148]	2006	0.27
Hybrid GA	Valls et al. [168]	2008	0.27
GA	Hartmann [152]	1998	0.27
PSO	Chen [144]	2011	0.29
SFLA	Fang and Wang [151]	2012	0.36
SAactivity list	Bouleimen and Lecocq [142]	2003	0.38
BA-FBI	Ziarati et al. [170]	2011	0.42
TS-activity list	Nonobe and Ibaraki [163]	2002	0.46

the PSPLIB data sets $J30$ and $J60$. The comparative study investigates performance of the GLSA against other state-of-art algorithms including genetic algorithm (GA),

Table 3.5: Average deviations from critical-path based lower bounds of $J60$

Algorithm	*Reference*	*Year*	Dev_{LB}
SFLA	Fang and Wang [151]	2012	11.44
Neurogenetic	Agrawal et al. [136]	2011	11.51
Hybrid GA	Valls et al. [168]	2008	11.56
Random Key GA	Mendes et al. [162]	2009	11.72
Scatter Search	Debels et al. [148]	2006	11.73
ACOSS	Chen et al. [145]	2010	11.75
GA	Alcaraz et al. [138]	2004	11.89
GLSA	**This study**	2013	**11.91**
ANGEL	Tseng and Chen [167]	2006	11.94
PSO	Chen [144]	2011	12.03
BA-FBI	Ziarati et al. [170]	2011	12.55
SAactivity list	Bouleimen and Lecocq [142]	2003	12.75
TS-activity list	Nonobe and Ibaraki [163]	2002	12.97
GA	Hartmann [152]	1998	13.30

simulated annealing (SA), tabu search (TS), bee algorithm (BA), ant colony optimization (ACO), particle swarm optimization (PSO), shuffled frog-leaping algorithm (SFLA) and their hybrids in view of the average derivation from the optimal solution. Table 5.1 lists the average deviation from the optimal makespan Dev_{avg} for the instance set $J30$ of all the algorithms respectively. GLSA obtained $Dev_{avg} = 0.19$. Hence, our proposed approach ranks 4th after the approaches of Mendes et al. [162], Agrawal et al. [136] and Chen et al. [145] for 1000 schedules.

Table 3.5 reports the average percentage deviation from the well-known critical path based lower bound (Dev_{LB}) for instance set $J60$. GLSA obtained $Dev_{LB} = 11.91$ and ranks 8th for 1000 evaluated schedules.

In the broad term, the obtain results confirm the effectiveness and competitiveness of GLSA in solving the RCPSP.

3.5 Conclusion

This chapter presents a hybrid metaheuristic algorithm based on genetic approach and local search algorithm to solve the RCPSP. The aim of this work is to improve the results provided by genetic algorithm (GA) by hybridizing it with a local search (LS) technique during the mutation process. The efficiency of the GLSA is ascertained by comparing the results with others proposed approximation algorithms in literature. For the data sets $J30$, GLSA is the fourth best in the comparison list (table 5.1) and for the $J60$ is the eighth best (table 3.5). From the viewpoints of both the solution quality and computational expenses, the proposed GLSA is efficient and effective method for solving the RCPSP.

This study mainly focuses on solving the classical well-known RCPSP with makespan minimization.Future investigations are required to be more attached to the reality by making our study more reliable when handling real life problems since in practice each company owns a large number of resources/assets/equipments to be shared between different projects and usually aims to accomplish the project/mission as soon as possible with the minimum cost and the maximum quality. Therefore, we investigate in the next chapter the assignment and scheduling problem as a multi-objective model and with multiple execution modes.

Chapter 4

A Genetic Approach for the Multi-Objective and Multi-mode Resource Allocation and Scheduling Problem

4.1 Introduction

In this chapter we address the multi-mode and multi-objective surveillance assets allocation and scheduling to different surveillance tasks. In particular, this problem can be represented as a Multi-Objective and Multi-mode Resource Allocation and Scheduling problem (MOMRASP). The current literature mostly addressed the single objective instance as a single objective optimization problem. The MOMRASP can be viewed as a two-step optimization: the assignment of patrolling resources (or modes) to accomplish tasks while respecting availability constraints and the scheduling of these tasks while satisfying precedence and priority constraints. Several objectives are to be considered as minimizing the mission makespan, total cost and maximizing

the probability of success. The MOMRASP is $\mathcal{NP}$-hard. Therefore, we propose a Multi-Criteria Genetic Algorithm ($MCGA$) based on multi-objective concepts to schedule a set of tasks while assigning the available and most suitable resources as the availability varies over time. The proposed approach explores the search space for finding a set of potential efficient solutions. The $MCGA$ reproduction procedure is based on multi-criteria and dominance concepts to select solutions. Crossover and mutation operators are applied to diversify the search space and improve the quality of solutions with respect to all objectives. The $MCGA$ is implemented in C++ language and run on a large set of randomly generated problems. $MCGA$ performances are compared to Ant System approach based on selected performance metrics. We also compute a lower bound for each objective in order to measure the degree of correctness of the obtained set of potentially efficient solutions. This chapter is organized as follows. The problem description is presented in section 2. Section 3 presents the mathematical formulation of the problem. The related contributions are reviewed in section 4. Section 5 presents the proposed $MCGA$ and the adaptations of its features. Section 6 reports the numerical experiments. The main conclusions are discussed in section 7.

4.2 Problem description

The large volume surveillance problem is characterized by the employment of mobile (such as helicopters, ships, airborne, etc) and fixed (as land radar, satellite, etc) patrolling and heterogeneous assets/resources to a large geographic area in order to accomplish the maximum number of surveillance tasks. Coastal and Arctic wide area surveillance are good examples of large volume SP. Thus, the surveillance mission consists of assigning a set of surveillance resources in order to achieve a set of surveillance and urgent tasks such as rescue of a fishing boat in distess or search of illegal immigrants.

In this research, we address the multi-mode and multi-objective surveillance assets allocation and scheduling to different surveillance tasks (MOMRASP). The problem

can be stated as a set of tasks to be scheduled, related by precedence constraints and where each task can be accomplished by one out of a set of different modes. Thus, the processing time, cost and probability to succeed the achievement of a given task depends on the adopted mode and it is considered as an endogenous variable that defines the combination of resources. Moreover, the duration includes the transition time of the resource. We assume that the resources are renewable and several objectives for the surveillance problem are to be considerered as minimizing the mission makespan, total cost and maximizing the probability of success.
The MOMRASP related to the large volume surveillance problem deals with finding:

❶ An efficient assignment of heterogeneous and networked set of surveillance assets.

❷ An efficient schedule for all surveillance tasks according to an imposed set of spatial and temporal constraints.

Figure (4.1) shows a precedence example, where the nodes identify tasks followed by their modes. Directed edges represent the precedence relationship. In figure (4.1), task 3, for example, can be accomplished by using either the first mode $\{R_2, R_3\}$ or the second one $\{R_4\}$.

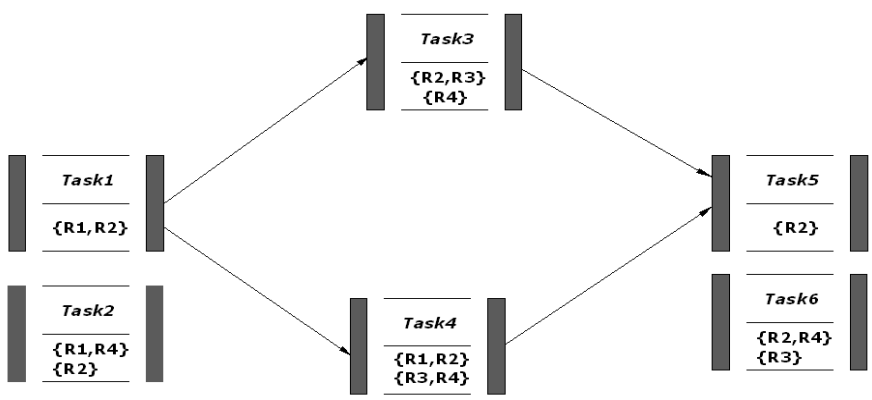

Figure 4.1: An illustrative example with $N = 6$ and $k = 4$

4.3 Problem formulation

The notation and parameters used in the mathematical model are summarized in table (4.1).

Objective Functions:

- The first objective function minimizes the total duration of the project. It is defined as the total time span between the starting and the finishing time of the project as follows in equation (1).

$$\textbf{Min } F_1(x) = Max \sum_{i=1}^{N} l_i \tag{4.1}$$

- The second objective function (2) minimizes the in-use costs of the resources required by all the tasks:

$$\textbf{Min } F_2(x) = \sum_{i=1}^{N} \sum_{m=1}^{M_i} \sum_{j=1}^{k} c_j \; x_{ijm}^{t} \tag{4.2}$$

- The third objective function is the probability of success. It is computed based on a priori probabilities that a given resource successfully achieves a given task to which it is allocated in specific conditions. The computation formula is shown by Equation (3):

$$\textbf{Max } F_3(x) = \frac{1}{N} \sum_{i=1}^{N} (\sum_{m=1}^{M_i} \sum_{j=1}^{k} P_{ij} \; x_{ijm}^{t} / \sum_{m=1}^{M_i} \sum_{j=1}^{k} q_{ijm}) \tag{4.3}$$

Constraints:

- Equations (4) ensure that the available quantity R_j of the renewable resources j available in period t, with $t = 1,\ldots,T_{max}$, will not be exceeded.

$$\sum_{i=1}^{N} \sum_{m=1}^{M_i} x_{ijm}^{t} \; q_{ijm} \leq R_j^t, \quad j = 1,\ldots,k \tag{4.4}$$

- Equations (5) guarantee that each task n_i starts if all its predecessors $n_p \in \hbar_i$ are executed:

$$s_i \geq Max_{p \in \hbar_i} (s_p + (\sum_{m=1}^{M_p} \sum_{j=1}^{k} \frac{x_{pjm}^{t}}{|m|} \; d_{pm})), \quad i = 1,\ldots,N \tag{4.5}$$

Table 4.1: Notations

N	Number of tasks
T	Set of tasks $\{n_i,\ i = 1,\ldots,N\}$
n_i	Task i
k	Number of resources
M_i	Number of modes that task i can be performed in
q_{ijm}	Quantity of resource of type j required to task i being performed in mode m
d_{im}	Duration of task i being performed in mode m
s_i	Starting time of task i at the beginning of the period t
l_i	Finishing time of task i with $l_i = s_i + d_{im}$
c_j	In use-cost of resource j
P_{ij}	Probability that resource j successfully realizes task i
$\|m\|$	Number of resources of mode m
$\hbar_i$	Set of predecessors of task i
R_j^t	Available quantity of resource of type j at the period t
r_j	Resource of type j
P_j	Probability of availability of resource j
a_i^m	Elementary action is the task n_i being realized by the mode m
T_{max}	Upper bound on the makespan
x_{ijm}^t	$\begin{cases} 1 & \text{If } n_i \text{ is realized by resource } j \text{ using mode } m \text{ at the period } t \\ 0 & \text{otherwise} \end{cases}$
y_{it}	$\begin{cases} 1 & \text{If task } n_i \text{ starts at the beginning of period } t \\ 0 & \text{otherwise} \end{cases}$

- Equations (6) impose that exactly one mode is assigned to each task:

$$\sum_{m=1}^{M_i} \frac{1}{|m|} \sum_{j=1}^{k} x_{ijm}^{t} = 1, i = 1,\ldots,N \tag{4.6}$$

- Equations (7) ensure that the preemption is not allowed:

$$\sum_{t=1}^{T_{max}} y_{it} \leq 1, \quad i = 1,\ldots,N \tag{4.7}$$

- Equations (8) and (9) are the binary requirements of the decision variables x_{ijm}^{t} and y_{it}:

$$x_{ijm}^{t} \in \{0,1\}, \quad i = 1,\ldots,N, \quad m = 1,\ldots,M_i, \quad j = 1,\ldots,k \tag{4.8}$$

$$y_{it} \in \{0,1\}, \quad i = 1,\ldots,N, \quad t = 1,\ldots,T_{max} \tag{4.9}$$

4.4 Review of the related work

To our knowledge, we are the first proposing to model the resource management for large volume surveillance as Multi-Mode and Multi-Objective Assignment and Scheduling Problem. Therefore, we focus the literature review mainly on the assignment and scheduling problem. Over the last decades, considerable efforts were dedicated to solve the assignment and scheduling problem (ASP), giving place to a large variety of models and formulations. This interest was motivated by many potential applications like software development, wireless systems and surveillance problems Grob (2006). Important characteristics and variants of the ASP were distinguished: single mode *vs* multi-mode, mono-objective *vs* multi-objective, with preemptive *vs* non-preemptive tasks, renewable vs non renewable resources and with regular *vs* non-regular objective, etc. Such variants have been investigated by Wang & al. (2001). The related literature can be viewed according to two main classes of solution approaches: exact and heuristic methods.
We propose to review each sub-class separetly, namely the resource assignment problem and the scheduling problem.

Exact methods: To our knowledge, the first exact method proposed for the multi-mode assignment problem was developed by Slowinski (1980). For solving the problem under resource availability constraints, the author proposed a two-stage linear programming approach. Speranza and Vercellis (1993) proposed a hierarchical two stage decomposition process for the nonpreemptive multi-project planning and scheduling problem. They developed a branch and bound algorithm. Hartmann and Sprecher (1993) have shown that this algorithm may not be able to find optimal schedule with two or more renewable resources and can not enumerate all the schedules and generally suboptimal solution can be found. Demeulemeester and Herroelen (1992) proposed a depth-first branch-and-bound algorithm to solve the single mode project scheduling problem under precedence and resource constraints where the objective consists in minimizing the makespan. Later, Sprecher et al. (1997) extended the proposed work of Demeulemeester and Herroelen by introducing mode alternatives. More recently, Heilmann (2003) proposed a branch and bound algorithm to solve the multi-mode resource-constrained project scheduling problem with minimum and maximum time lags such that the makespan was minimized. Zhu et al. (2006) proposed a branch and cut algorithm. Ng and Sancho (2009) modeled the surveillance of a sequence of regions to be searched by one maritime patrol aircraft as mono-objective travelling salseman problem with the minimization of the mission time as an optimization objective function. A dynamic programming formulation was proposed for handling the regional surveillance problem. Liu et al. (2006) studied the maximal lifetime scheduling problem for k to 1 sensor-target surveillance system. They proposed an optimal solution method to schedule sensors to observe target such that the lifetime of the system is maximized. The problem was addressed in three steps: first, they computed the maximal lifetime of the system based on linear programming method; second, they decomposed a workload matrix into a sequence of schedule matrices by using perfect k-matching method and finally, they obtained target watching timetable for the surveillance sensors. These mathematical optimization methods are considered powerful techniques to find optimal solutions for sequential decision problems by finding the suitable decision for all possible states at each stage. However, for large-scaled and multi-objective decision problems, none of these procedures can be

used since the solution space is too large and they are not able to generate an optimal solution in reasonable computational time. Therefore, it is necessary to adopt better methodologies to search the efficient solutions space that satisfy the decision maker in reasonable time. Heuristic methods appear to be powerful solution techniques for handling large-sized decision problems.

Heuristic approaches: ASP is $\mathcal{NP}$-hard, many heuristics were proposed to solve the problem in reasonable computation time. The related approximation approaches can be classified on three main categories: Dispatching rules, Artificial Intelligence(AI)-based approaches and Agent-based approaches. Dammak et al. (2006) studied the problem of assigning a set of independent exams to a set of classrooms with limited capacity. They formulated it as 0-1 linear integer program. The problem consists only in satisfing two sets of constraints: each exam should be assigned to a classroom having total capacity greater than the number of students, and no more than one exam in each classroom. They used a simple heuristic procedure called max-size assignment to solve the exam-classroom assignment problem. The proposed approach assigns exams to classrooms in such a way that the two constraints are satisfied. If no feasible assignment was found they relax the second constraint of the problem and allow to have two exams in the same classroom. However, the heuristic can fail to find a feasible assignment even with the relaxation of the second constraint. Lee and Lee (2005) proposed a hybrid search approach based on genetic algorithm and ant colony optimization to solve the mono-objective resource allocation problem. The genetic algorithm was used to provide the diversity in search space and ant colony was performed to exploit best solution. The addressed problem formulation is based on two assumptions: all the resources must be assigned to activities; and an individual allocating probability P_{ij} for the fulfill of the ith activity by assigning the jth resource is known for all i and j. They tried to minimize the penalty of the ith activity not being fulfilled by its assigned resource. Belfares et al. (2007) proposed a progressive resource allocation method based on the tabu search heuristic and a posteriori approach to solve the multi-objective RCPS problem with multiple modes. They used a Multi-criteria Filtering Procedure and the Dominance Rule to filter and to choose the efficient solutions. For the assignment problem, we refer the reader to Chaharsooghi

and Kermani (2008), Dai et al. (2003), Korhonen et Syrjänen (2004) and Lin and Gen (2008). Mendes et al. (2009) proposed a random key-based genetic algorithm for the single mode resource constrained project scheduling problem by adopting an indirect representation of the chromosome solution. The generated chromosomes do not represent the solutions to the original problem and special concepts can be used to derive the solutions. They classified the schedules into three types. First type is the semi-active schedules: where feasible schedules are obtained by sequencing activities as early as possible. No activity can be started earlier without changing the sequence. Second, the active schedules: in which no activity can be delayed without delaying some other activity or breaking a precedence relationship. Optimal schedules are always members of this set of schedules and active schedules are members of the set of semi-active schedules. Third, the non-delay schedules: in which no resource is allowed to be idle when it could start to process an activity. Non-delay schedules are also members of the set of active schedules. In their work, Mendes et al. (2009) limited the search space in the set of active schedules. However, the active schedules space is very large and beside containing the optimal schedule, it usually contains large number of solutions with poor quality in terms of makespan. They used, the concept of parameterised schedules to reduce this search space by placing a restriction on the project duration and to generate the direct solution. They employed a unique fitness measurement criterion termed modified makespan to differentiate between schedules having the same makespan. Alcaraz and Maroto (2001) proposed a self-adaptation of the genetic algorithm to solve the single mode resource constrained project scheduling problem in such way the makespan was minimized. The initial population was not generated randomly but they used a sampling method based on the Latest Finish Time as the selection priority rule. They added an additional gene into the solution chromosome in order to control the way in which the phenotype was built. This gene called forward/backward gene, which determines whether forward or backward pass scheduling has been employed. Loukil et al. (2005) proposed a Multi-objective simulated annealing method to solve the multiple objectives production scheduling problem. They presented the model as sequences of permutation of jobs in each machine and they used three way to define the neighbor of a sequence:

first, two jobs are exchanged; second, a job is inserted and the third, the neigbhor was chosen randomly. They used a weighting function for the determination of a good direction on the efficient frontier.
Kolish and Hartmann (2000),(2006), Brucker et al. (1999) and Herroelen et al. (1998) published different surveys of the heuristic approaches for solving the scheduling problem.
The ASP is formulated either as an asignment or a scheduling problem. We propose to formulate the MOMRASP as the assignment of modes to tasks and the scheduling of tasks to be accomplished, giving rise to an assignment and scheduling problem as shown in equations (1-9). Given that it is a $\mathcal{NP}$-hard problem, we consider evolutionary algorithms for solving the MOMRASP. Given their efficiency to exhibit good approximation of the pareto optimal sets while requiring reasonable computational time. These algorithms generally beat traditional mathematical programming methods when the ratio -quality of discovered potential optimal solution sets/ time consumption in both development and implementation- is considered. In this context and facing the multiplicity and the complexity of both objectives and constraints, we propose a multi-criteria genetic algorithm as our resolution strategy that simultaneously Pareto-optimize the three objective functions by using a selection operator based on dominance and multicriteria filter. The basic feature of the $MCGA$ is the multiple directional and global search, which allow us to find several solutions of the pareto optimum set due to the maintenance of the potential solutions progressively.

4.5 Genetic Algorithm: background

The GAs were inspired by natural biological evolution. GAs were developed based on the Mendel's rules and the fundamental principle of Charles Darwin and have been developed and proposed by John Holland at the university of Michigan in 1970's [33]. Contrarily to TS, which is considered as sequential search method starts from an initial feasible solutions and attempt to improve it by moving to the neighbor of the solution, GA is a stochastic search method based on the mechanisms of natural se-

lection and natural genetics in biological evolution [25]. This method work with a random population of solutions. GAs have been applied to a wide variety of combinatorial optimization problems and have been proved their efficiency. The first step of the optimization process using GA is to code the parameters of the objective functions. Then, solutions are represented as an array or a string, called *chromosome*. Each chromosome is evaluated and measured by a fitness functions. The improvement of the solutions at each generation is done by using the principle of the fittest process.

4.6 A MCGA for the MOMRASP

The $MCGA$ is proposed to generate the potential optimal solution sets for the MOMRASP by finding an efficient assignment of the set of patrolling resources to tasks and an efficient and feasible surveillance schedule for all surveillance tasks. The merits of the different solutions is assessed by considering the best compromise between the different decision criteria. The principle of non-dominance and Pareto-optimality concepts are used during the optimization process and the outcome of the algorithm is stored in an external file. The proposed resolution strategy is detailed in what follows.

4.6.1 Encoding variables

A chromosome represents a solution to the problem encoded as a random vector. The direct representation for the assignment and scheduling problem is complex and the indirect one needs special procedures to deduce a solution. The representation used in this work is a direct encoding. Each solution chromosome is made of $3N$ genes, where N is the number of tasks.

$$chromosome = (\underbrace{gene_1, \ldots, gene_N}_{mode}, \underbrace{gene_{N+1}, \ldots, gene_{2N}}_{period}, \underbrace{gene_{2N+1}, \ldots, gene_{3N}}_{priority})$$

The first N genes are used to determine the mode assigned to each task. The second N genes are used to determine the period of each task. The third N genes are used

to represent the priority of each task.

mode			*period*			*priority*		
$gene_1$	...	$gene_N$	$gene_{N+1}$	...	$gene_{2N}$	$gene_{2N+1}$	...	$gene_{3N}$

Encoding of the mode

The first N genes are used to obtain the combination of resources or mode assigned to each tasks. Each mode has different value of cost and processing time. The mode of each task i is given by the gene value:
$mode_i \ = \ gene_i$

Mode		
$gene_1$	...	$gene_N$

Encoding of the period

The genes between positions $N+1$ and $2N$ are used to determine the period in which the correspond task will be realized by taking into account the precedence constraint between tasks. The period of each task i is given by the gene value:
$period_i \ = \ gene_{N+i}$

Period		
$gene_{N+1}$	...	$gene_{2N}$

Encoding of the priority

The genes between positions $2N+1$ and $3N$ are used to represent the tasks priorities. The tasks priorities value are positive integers. The higher the value, the higher the priority will be. The priority of each task i is given by the gene value:
$priority_i \ = \ gene_{2N+i}$

Priorities		
$gene_{2N+1}$	...	$gene_{3N}$

Fitness functions

The proposed approach is based on the concept of non-dominated solutions. The objective functions defined in the mathematical model $(1)-(9)$ will be used as fitness mesures for each chromosome solution.

4.6.2 Evolutionary operators

To explore the solution space and to breed good quality chromosome solutions, three combination operators are applied: mutation, recombination (crossover) and reproduction (selection). Evolution of the population from one generation to another depends on the power of these operators. There are many variants of the GA due to the varations of the reproduction, crossover and mutation operators. The reproduction and the crossover operators determine which parents will be selected, and how genetic material (genes) will be exchanged between parents to create next generation (offspring). The two operators contribute to perform the quality of population and tend to convergence. Mutation operator used to diversify the solutions and contribute to produce new points in the solution space.

Selection operator

The selection or reproduction operator consists of retaining the best individuals from the current population and copied them unchanged into the next generation based on their fitness value. This selection method is called elitist or elitism (Goldberg (1989)). It forms a succesful selection strategy used to ensure that the best solutions are preserved in the next generation and allows to converge towards the pareto frontier.

Crossover operator

Two individuals are randomly selected from the current population to act as parents. For each gene a random number between [0,1] is generated. If the generated number is smaller than a threshold value, the gene of the first parent is copied into the offspring chromosome. Otherwise, the gene of the second parent is used. The threshold value is an input data and is called *Crossover Probability*.

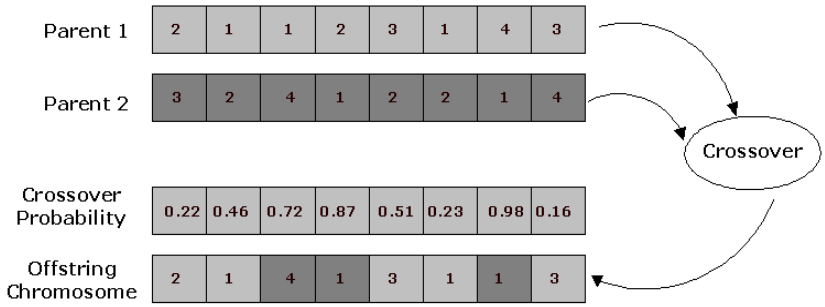

Figure 4.2: The crossover operator

As reported in figure (4.2), the crossover of the parents 1 and 2 using the probability threshold $\alpha = 0.7$ operates as follows: if probability $< \alpha$ the heritage is from parent 1, otherwise the heritage is from the second parent. Hence, using the crossover probability array, the offspring chromosome is generated.

Mutation operator

Randomly applied to explore other areas in the solution space and avoid the convergence caused by selection and crossover operators. The probability of the mutation M_r is inversely propotional to the population size. Thus, after the crossover has occurred, an individual can be selected from the current population for mutation. It consists to switch the mode associated to the selected task i based on its neighborhood set $\aleph_i$ of the resources' combination.

4.6.3 The algorithm

The parameters used in the GA are: the population size (pop_{size}), the number of generations (G), the crossover probability ($cross_p$) and the mutation rate (M_r).

Initialization

- Generate an initial population P_0 of pop_{size} solutions or chromosomes;
- Initialize the crossover probability $cross_p \leftarrow \alpha$;
- $P_G \leftarrow P_0$;
- Evaluate each chromosome by calculating its fitness function;
- $g \leftarrow 0$;

Iterative Process:

- At iteration g check if $g > G$, stop
- Else $g \leftarrow g + 1$ go to step 1
- Step 1: Select randomly two chromosomes c_1 and c_2 from P_G;
- Step 2: At iteration i Generate random number $\beta \in [0, 1]$;
- If $\beta \leq \alpha$: $offspring \leftarrow allele(c_1)$ otherwise $offspring \leftarrow allele(c_2)$;
- Step 3: Check the activation of the mutation, If yes: Mutate some alleles;
- Else go to step 4;
- Step 4: Check feasibility then filtrate the new population;
- Step 5: $P_G \leftarrow P_G \cup P_i$;
- Step 6: Check if $i > pop_{size}$, stop, Else $i \leftarrow i + 1$ go to step 1;

The proposed algorithm is illustrated as shown in figure (4.3). The $MCGA$ starts by generating randomly the initial population and the crossover probability. The iterative process consists in selecting two chromosomes as parents. Subsequently, the crossover operator is applied to generate an offspring that contains a combination of genes from its parents. The activation of the mutation operator, which consists to

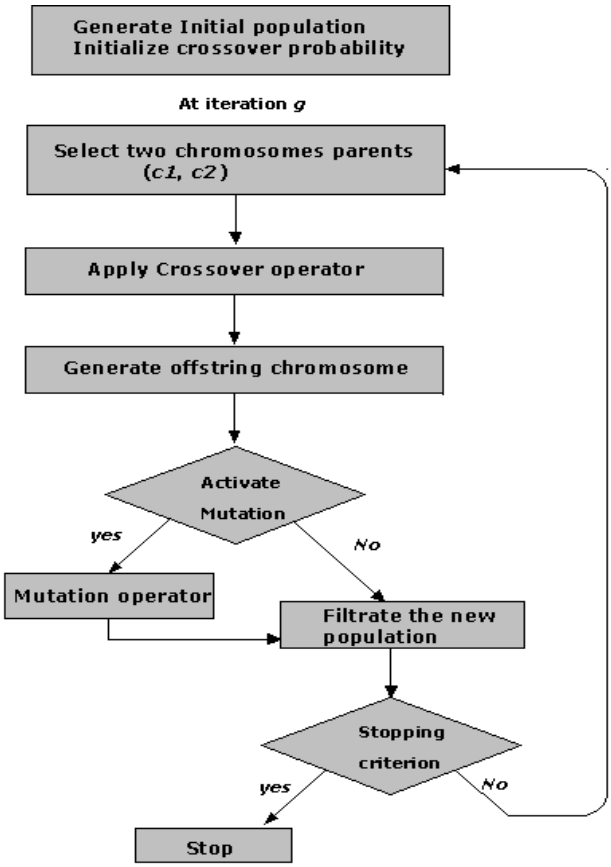

Figure 4.3: The algorithm

change some alleles, is then checked. The process is enclosed by a check of the new generated solutions.

4.7 Computational experiments

We report the *MCGA* performance based on the diversification characteristics, computing time and deviations of the generated solutions from lower bounds. The results were generated using a Pentium IV 2.53 GHz processor with operating system Microsoft Windows XP and 512 MB RAM. The algorithm was coded in Visual C++ 6.0.

Table 4.2: The configuration of the parameters

Configurations	$\mathbf{C_1}$	$\mathbf{C_2}$	$\mathbf{C_3}$
Parameters		*Values*	
pop_{size}	N	$2N$	$3N$
$cross_p$	0.6	0.7	0.8
G	100	150	200

4.7.1 Parameter settings

Before running the GA, we try to initialize the input parameters such as the population size pop_{size}, generating the initial population P_0, defining the number of generations G, the crossover probability $cross_p$ and the mutation rate M_r:

- The population size is an important parameter of the GA and can affect the efficiency and performance of the algorithm in terms of convergence and computational time. A small population size can provide insufficient possibilities to perform crossover operator and to exploit the solution space, however, a large population require more processing time. Therefore, the population size can be proportional to the problem size. We varied the pop_{size} of the population from N to $3N$.

- The initial population is randomly generated with a test of feasibility of the schedules.

- The number of generations is drawn in {100,150,200}.

- The probability of crossover $cross_p = (0.6, 0.7, 0.8)$.

- The mutation rate M_r=$\frac{1}{pop_{size}}$ which is give a better result compared to fixed value.

Table (4.2) presents the three configurations used to generate the solution set.

In an attempt to obtain reasonable and effective inputs, we generated a series of tests to fine tune the different parameters of GA. Table (4.3) reports the numerical

results for different configurations depending on the problem size.
The first column of table (4.3) represents problems instances. The second column refers to the results obtained according to configuration C_1 and it shows the best fitness functions found for F_1, F_2 and F_3 respectively. Based on the computational results displayed in table (4.3), we can notice that the configuration C_2 generates $(23\%, 42\%, 23\%)$ of best solutions for the three objective functions respectively. However, C_1 and C_3 generate $(6\%, 6\%, 11\%)$ and $(6\%, 6\%, 52\%)$ of best solutions for all objectives. Since the configuration C_2 obtains best results, it will be uesd in our algorithm for the experimental tests.

Table 4.3: Algorithm performances with C_1, C_2 and C_3

Configurations	$\mathbf{C_1}$			$\mathbf{C_2}$			$\mathbf{C_3}$		
Pbs	F_1	F_2	F_3	F_1	F_2	F_3	F_1	F_2	F_3
P_1	27	122	35.5%	26	122	35%	26	122	35.7%
P_2	12	192	87.7%	11	172	91%	12	165	93.5%
P_3	47	133	38%	47	133	41.1%	47	133	39.5%
P_4	27	131	35.5%	26	131	35.3%	26	131	35.7%
P_5	31	77	89%	27	68	86.6%	30	77	86.6%
P_6	63	845	33.1%	63	793	31.2%	63	794	32.3%
P_7	27	122	35.5%	26	122	35%	26	122	35.7%
P_8	180	1384	29.5%	178	1369	30.5%	178	1369	35.7%
P_9	372	1592	27%	355	1562	34%	354	1562	31.7%
P_{10}	52	1059	24.5%	53	1053	25.9%	52	1056	26.7%
P_{11}	120	1846	31.3%	120	1783	31.8%	120	1803	32.2%
P_{12}	475	1687	22.5%	443	1679	23.6%	444	1679	22.8%
P_{13}	678	2866	27.6%	676	2767	29.9%	678	2767	30.7%
P_{14}	653	3219	28.6%	649	3186	29%	649	3199	29.3%
P_{15}	323	2987	30.7%	326	2988	31%	330	2988	30.8%
P_{16}	317	5640	36.5%	317	5626	37%	317	5636	37.6%
P_{17}	427	7693	36%	427	7556	35.7%	427	7592	36.1%
Number of best	7	4	2	14	16	4	12	10	11

4.7.2 Experimental results

Table (4.4) summarizes the test results for test instances taken from Ben Abdelaziz et al. (2007). Let $Avg_{N.P}$ denote the average number of predecessors for each task. The table lists the CPU time for each instance. The results show that $MCGA$

outperforms AS and take relatively short time then the AS. The execution time spended by $MCGA$ varies from 0.06 to 1.48 seconds, however, for the AS varies from 0.13 to 4.63 seconds.

Figure (4.4) summarizes the CPU time's variation in accordance with the problem sizes for the two approaches. It is noticeable that AS is more time consuming than $MCGA$. It is not surprising that $MCGA$ converges faster to the pareto-optimal front than AS, which belongs to the class of constructive methods.

Table (4.5) drives the average deviations from the critical path lower bound Gap_{F_1} (Kolish and Hartmann (2000) and Patterson and Huber (1974)), the lower bound of the cost Gap_{F_2} Ben Abdelaziz et al. (2007) and the lower bound of the probabilily of success Gap_{F_3} for different instances size. As illustrated in table (4.5), we can see that the deviations from the lower bound for the three objectives is relatively short compared to AS, thus, the average gap for $MCGA$ is about $(0.09, 0.08, 0.08)$ for the three objective functions F_1, F_2 and F_3 respectively, however, for the AS is about $(0.12, 0.12, 0.08)$. It is noticeable that for problems P_1, P_3, P_4 and P_5 $Gap_{F_i} = 0$ for some $i \in \{1, 2, 3\}$. The solutions found concide with the best lower bound known.

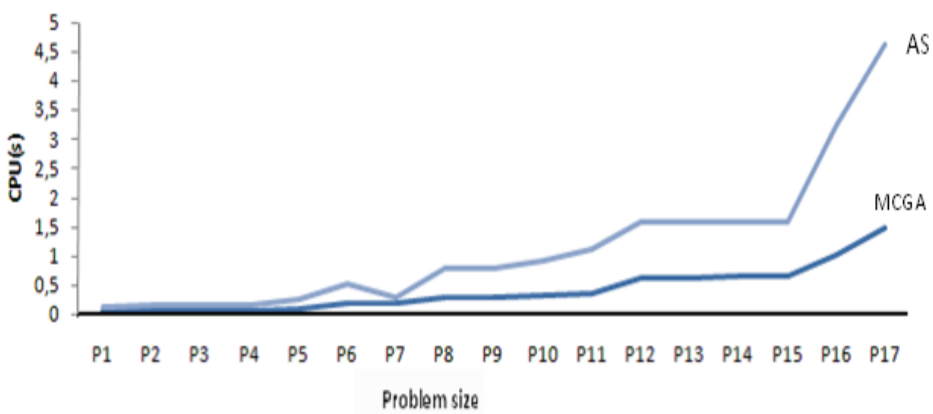

Figure 4.4: The CPU time for problems P_1-P_{17}

4.7.3 Diversification of the approximation set based on the diversity indicators

We propose three indicators to examine the diversification of the solutions set: the cardinality of the approximation set N_s, the diversity of the solutions in the decision space D_s and the spread of solutions on the pareto front Cov.

- *Cardinality of the approximation set*:
  The approximation pareto set is represented by the non-dominated solutions generated by the approach and stored in file or list. This later is updated each time the selection procedure is processed. An approximation set with higher value of N_s is preferred to a set with lower value, as the solutions will be used for a further classification.

- *Diversity of the approximation set*:
  The divesification of the population through the exploration and exploitation operators produces new points in the solution space. Ultimatly, the diversity of the solution in the pareto set represents a benefical result as it can give to the decision maker a larger range of options. The diversity of the population used here is inspired by Shannon's theory (Grünwald and Dawid, 2004) to evaluate the value of information based on the entropy H. This later is a mathematical function that corresponds to the amount of information delivered by a source of information and where $H(p) = -\sum_x p(x)log(p(x))$. The higher the entropy of a message ($0 \leq H \leq 1$), the higher the value of this message is. Thus, the entropy represents the extra information or the information rate (How much information we gain).
  We define the diversity of a solution as a variation of its modes.

$$D_s = \frac{1}{N}\sum_{i=1}^{N} H_i \tag{4.10}$$

With

$$H_i = -\frac{1}{log(m_i)} \sum_{j=1}^{m_i} \frac{n_{ij}}{pop_{size}} log(\frac{n_{ij}}{pop_{size}}) \tag{4.11}$$

Where D_s is the diversity of the population, H_i is the entropy of the task i, N is the number of tasks, m_i is the number of modes of task i and n_{ij} is the number of solutions with different modes for task i. So, the higher the number of solutions with different modes per task, the higher the diversity of solutions set is.

- *Diversity of the pareto approximation front*:
  We propose to express the diversification of the solutions spread all over the pareto frontier by the extent of coverage in each objective's dimension separately Wu and Azarm (2001). The extent of coverage relative to each objective F_i is defined as follows:

$$Cov_{F_i} = \frac{max_{x,y \in P_{known}} |F_i(x) - F_i(y)|}{|F_i^* - F_{i*}|} \times 100, \quad i = \{\ 1,2,3\} \tag{4.12}$$

  Where the numerator expresses the maximum distance among solutions of the approximation set, P_{known}, in the dimension i, and F_i^* and F_{i*} are, respectively, the ideal and anti-ideal values of F_i.

As illustrated in table (4.5) and based on the quality of discovered potential efficient solutions set, the generated solutions by $MCGA$ are better in terms of diversity and cardinality indicators compared to AS. The diversification of the solution in the pareto set for P_2 is about 91%, however for the AS is about 76%, which indicates that $MCGA$ explores better the search space. Furthermore, for small size problem $MCGA$ provides solutions with higher diversity and coverage rate in the objectives space. However, for large problems as P_{17} the diversification of the solution set D_s is about 90% of the diversity of the solution in the pareto set, but, the coverage in the objectives space is not too higher compared to small problem. We considered a sample of 17 MOMRASP instances and computed the coverage metric for each instance (see equation (4.12)) for both MCGA and AS on all the objectives. Table (4.6) is

a comparison of the two approaches. It is noticeable from table (4.6) that MCGA outperforms AS for each objective F_i $(i = 1,\ 2,\ 3)$ in most instances (64.7%,58.8% and 82.3%) of diversity of the pareto approximation front and subsequently, AS outperforms MCGA in fewer cases. Figures (4.5) clearly illustrate that $MCGA$ generates a better coverage of the approximation pareto frontier for all objectives.
We applyed the Wilcoxon signed-rank test to get a statistical significance of the results obtained by the two approaches. The choice of Wilcoxon test is motivated by the fact that the data distribution are not symmetrical. Based on the results displayed in table (7), we can conclude that there is a significance evidence that $MCGA$ is the best in terms of the diversity of the pareto approximation front for $Cov_{\{F_1,F_3\}}$ and the cardinality of the approximation set N_s then AS at 95.5% confidence level. However for Cov_{F_2} and the diversity of the approximation set D_s we cannot consider the obtained results by the two approaches to be significantly different.

4.8 Conclusion

In this chapter, we addressed the resource management for large volume surveillance problem modeled as a multi-objective and multi-mode resource allocation and scheduling problem defined through a finite set of surveillance resources and tasks. We started by stating the multi-objective mathematical formulation for the studied problem which optimize concurrently: the mission makespan, total cost and probability of success. Given the $\mathcal{NP}$-hardness of the problem, we developed a $MCGA$ as our resolution approach based on multi-criteria and non-dominance concepts. The proposed approach explores the search space for finding a set of potential efficient solutions by applying efficiently their combination operators. The merits of the generated solutions was assessed by considering the best compromise betweeen the different decision criteria.
The approach was tested on a large set of randomly generated problems. $MCGA$ performances are compared to AS approach based on selected performance metrics.

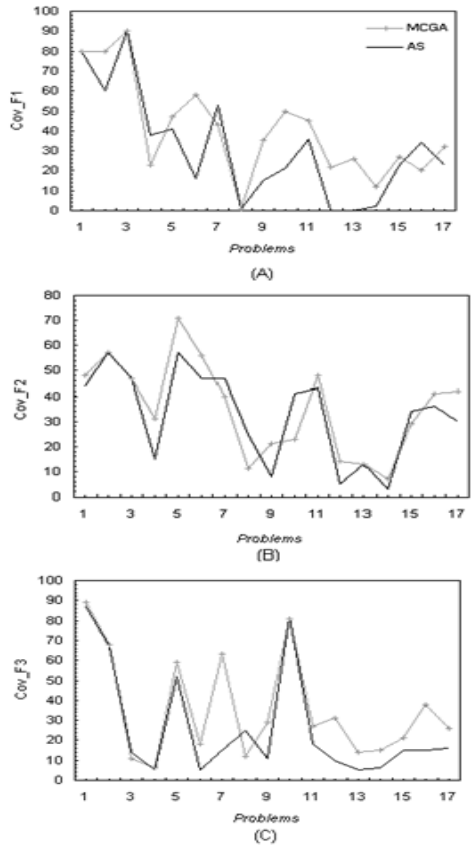

Figure 4.5: Diversity of the pareto approximation front for problems P_1-P_{17}

The empirical results experimental show that $MCGA$ outperforms the AS for most of the instances. Furthermore, we computed a lower bound for each objective and for the different instances. We noticed that the set of potential efficient solutions is comparable with these lower bounds for the two approches. The averge Gap of the deviation from the lower bounds is relatively small for the $MCGA$ then the AS. We found also that the $MCGA$ takes relatively short time then AS for all the tested instances. The favorable results obtained can validating the effectiveness of the proposed algorithm. However, should not be used to indicate that the $MCGA$ is always the better one. In fact, the comparison made shown that AS outperforms in fewer instances the $MCGA$ on the diversity indicators.

This work will be extended to consider more realistic assumptions of the resource management problem related to the large volume surveillance problem, such as fuel constraints, time windows and travel time. As results, the extended model will become computationally intractable for large sized problem when adding some realistic assumptions. As we have addressed simultaneously assignment and scheduling problem, we propose in the next chapter a rigorous bi-level decomposition model that reduces the computational effort of the problem by decomposing it into an upper and a lower level problem.

Table 4.4: Test results for different problem sizes

Problems	N	k	$\mathbf{Avg_{N.P}}$	$\mathbf{CPU(s)_{MCGA}}$	$\mathbf{CPU(s)_{AS}}$
P_1	6	3	(1-2)	0.06	0.13
P_2		4	(1-2)	0.06	0.16
P_3		5	(1-2)	0.06	0.16
P_4		5	(2-3)	0.06	0.16
P_5	11	12	(1-3)	0.09	0.27
P_6	30	4	(0-2)	0.19	0.52
P_7		4	(0-4)	0.19	0.30
P_8	50	3	(1-2)	0.29	0.80
P_9		3	(2-3)	0.30	0.80
P_{10}		10	(1-2)	0.32	0.91
P_{11}	60	4	(1-2)	0.35	1.11
P_{12}	100	3	(1-2)	0.61	1.58
P_{13}		3	(2-3)	0.61	1.58
P_{14}		5	(2-3)	0.65	1.58
P_{15}		15	(1-2)	0.67	1.59
P_{16}	150	15	(1-2)	1.03	3.23
P_{17}	200	20	(1-2)	1.48	4.63

Table 4.5: Computational results of the *MCGA* and *AS* appraoches

	MCGA						**AS**					
Pbs	Gap_{F_1}	Gap_{F_2}	Gap_{F_3}	N_s	D_s	$Cov_{(F1,F2,F3)\%}$	Gap_{F_1}	Gap_{F_2}	Gap_{F_3}	N_s	D_s	$Cov_{(F1,F2,F3)\%}$
P_1	0	0.1	0.01	9	0.85	$(80, 48, 89)$	0	0.12	0.02	5	0.78	$(80, 44, 87)$
P_2	0.01	0.03	0.01	16	0.91	$(80, 57, 68)$	0.03	0.03	0.01	10	0.76	$(60, 57, 67)$
P_3	0	0	0.05	17	0.76	$(90, 47, 11)$	0.03	0	0.02	10	0.78	$(90, 47, 14)$
P_4	0.06	0	0.07	8	0.51	$(27, 31, 6)$	0.04	0	0.02	6	0.57	$(32, 15, 5)$
P_5	0.03	0	0	18	0.58	$(47, 71, 59)$	0.23	0.13	0	14	0.78	$(41, 57, 52)$
P_6	0.04	0.01	0.07	20	0.94	$(58, 56, 18)$	0.08	0.13	0.14	10	0.92	$(16, 47, 5)$
P_7	0.2	0.01	0.08	18	0.58	$(43, 40, 63)$	0.2	0.1	0.1	11	0.78	$(53, 47, 15)$
P_8	0.1	0.14	0.07	17	0.7	$(1, 11, 12)$	0.18	0.12	0.07	7	0.67	$(1, 25, 25)$
P_9	0.13	0.1	0.05	13	0.52	$(35, 21, 29)$	0.13	0.12	0.08	7	0.45	$(15, 8, 11)$
P_{10}	0.04	0.1	0.1	15	0.75	$(50, 23, 81)$	0.04	0.13	0.09	9	0.81	$(21, 41, 82)$
P_{11}	0.04	0.11	0.1	44	0.88	$(45, 48, 27)$	0.27	0.05	0.11	23	0.91	$(36, 43, 18)$
P_{12}	0.2	0.13	0.12	8	0.31	$(22, 14, 31)$	0.3	0.15	0.13	5	0.19	$(0, 5, 10)$
P_{13}	0.2	0.07	0.1	12	0.23	$(26, 13, 14)$	0.2	0.3	0.12	4	0.16	$(0, 13, 5)$
P_{14}	0.3	0.2	0.14	16	0.27	$(12, 7, 15)$	0.3	0.25	0.2	8	0.21	$(2, 3, 6)$
P_{15}	0.08	0.13	0.13	43	0.86	$(27, 29, 21)$	0.1	0.14	0.13	11	0.62	$(23, 34, 15)$
P_{16}	0.09	0.19	0.14	57	0.89	$(32, 41, 38)$	0.09	0.2	0.14	14	0.68	$(34, 36, 15)$
P_{17}	0.08	0.18	0.14	64	0.90	$(32, 42, 26)$	0.08	0.17	0.14	11	0.72	$(23, 30, 16)$
$\mathbf{Avg_{Gap}}$	0.09	0.08	0.08				0.12	0.12	0.08			

Table 4.6: Comparison between MCGA and AS approaches based on Cov_{F_i} metric for P_1-P_{17}

		MCGA	**AS**
MCGA	F_1	–	64.7%
	F_2	–	58.8%
	F_3	–	82.3%
AS	F_1	17.6%	–
	F_2	23.5%	–
	F_3	17.6%	–

Table 4.7: p-values of the MCGA vs AS comparisons

Diversity indicators	N_s	D_s	Cov_{F_1}	Cov_{F_2}	Cov_{F_3}
p-value	$0,003$	$0,2906$	$0,0466$	$0,0804$	$0,0472$

Chapter 5

An Evolutionary Algorithm for the Bi-level Multi-platform Large Volume Surveillance Problem

5.1 Introduction

In this chapter, we investigate the scheduling and assignment of heterogeneous surveillance assets to surveillance tasks in a continuous and connected space. Surveillance tasks are characterized by their dependencies, priorities, as well as time and space constraints. In addition, tasks can be performed according to several modes where each mode can be characterized by a task-time duration, resource consumption, cost and benefits. Surveillance assets require transition time between tasks and some time a release time.Thus, the problem seeks to find the optimal assignment of a fleet of surveillance resources to a set of surveillance tasks as well as the corresponding sequencing scheme of tasks over a time period. In broad terms, given a set of tasks, a set of resources, a set of constraints and a set of predefined objectives, the main question is how should the resources be assigned to accomplished tasks and how these

tasks should be scheduled over time?

Answering this question, however, usually requires complex time-costly procedures. To address these computational difficulties, strategies based on aggregation, decomposition, excat and heuristics have been considered in literature. Decomposition techniques on the other hand are generally based on two-level decomposition schemes. As the problem can be viewed as a combined optimization problem seeking to find the optimal schedule and resources assignment and due to interdependencies between assignment and scheduling phases, numerous trade-offs between decisions can be managed. Since, the interplay between these two processes can be viewed as a sequential decision process. In this context, the main aim of this chapter is to design and develop a novel bi-level model for the ASP. The proposed method adopts a hierarchical bi-level search framework. At the upper level decisions for sequencing the set of surveillance tasks over a time period, such that the makespan is minimized, are made. On the other hand, at the subordinate lower level the assignment of the surveillance resources is addressed with the objective to minimize the total cost.

At present, the various algorithms developed for solving the BLP can be classified into two main categories: exact algorithmic methods and metaheuristic methods. Exact methods involve the use of enumerative scheme, whereas others replace the second level problem with the Karush-KuhnTucker (KKT) optimility conditions, yielding into a nonlinear one-level problem with complementarity constraints. However, these algorithms are far from being efficient in terms of computational time involved when solving large problems. Metaheuristic algorithms have proved to be robust in finding good solutions to complex optimization problems in reasonable computational time. In this context and given that the ASP is NP-hard, the use of an exact approach is inconceivable for the targeted large-scale instances. Therefore, the proposed method is an approximate algorithm. It alternates between a sequencing phase, solved by a hybrid genetic algorithm (HGA), and a resources assignment phase, handled by a GLSA.

The objective of this work is to propose new evolutionary bi level approach for the ASP. More precisely, we make the following contributions:

❶ We propose to formulate the ASP as a bi-level optimization problem

❷ We develop a new bi-level optimization approach to handle the proposed model

❸ We present the results of the comprehensive computational study while requiring a modest computing time.

The remainder of this chapter is organized as follows. In section 2..

5.2 Problem description

The bi-level assignment and scheduling problem (BLASP) is characterized by:

- ***A vector of surveillance tasks:***
  $T = \{1, \ldots, n\}$ for each task i we have the following attributes:

  - time window representing the earliest starting time and latest ending time $[a_i, b_i]$.
  - spatial localization (X, Y).
  - processing time that dependent on the used mode p_{im}.
  - Each task i requires one or more resources and can be performed according to several modes depending on the available quantity and ability of resources required to accomplish the task i.
  - set of predecessors
  - priority order.

- ***A pool of surveillance resources/assets***
  A set of renewable R resources with the following attributes:

  - available quantity during an interval of time.
  - localization of the resource.

- type and ability of resource.
- specific characteristics: in-use cost, speed, probability to succeed the accomplishment of given task, ...
- need fuel energy: automomy.

- ***A set of constraints***
  The BLASP includes various forms of constraints differing in difficulty and complexity which makes the problem extremely hard to solve. The constraints must be satisfied because they define the feasibility of a schedule. The constraints of the problem can be classified as task constraints and resources constraints.

```
- precedence relationship
- priority of tasks
- availability of resources
- fuel constraints
```

- ***A set of conflicting objectives***
  The large volume surveillance problems are subject to multiple, often conflicting, objectives that should be simultaneously optimized. The BLASP should be designed to optimize this typical objectives:

```
- mission makespan
- total cost
```

5.3 Mixed-integer linear formulation

5.3.1 Notations

The different used notations are discribed as follows:

Index sets

- i: the task index, $i \in \{1, \ldots, N\}$

- m: the mode index, $m \in \{1, \ldots, M_i\}$
- R: the set of renewable resources

Parameters

- M_i: number of mode that task i can be perform in
- E_k: constant amount of renewable resource $k \in R$ is available at any time
- p_{im}: processing time of task $i \in N_1 \subset N$ being performed using mode m
- d_i: a fixed processing time of task $i \in N_2 \subset N$
- $[a_i, b_i]$: time window on the completion of task i
- T: an upper bound on the makespan
- P: Precedence relations: a set of pairs such that $(i, j) \in P$ means that task i precedes task j
- r_{imk}: number of units of the renewable resource $k \in R$ required by task i using mode m
- c_k: the cost of using the renewable resource $k \in R$
- l_{ik}: travel time of the renewable resource $k \in R$ to reach task i: $l_{ik} = D_{ik}/S_{ik}$ where D is the distance and S is the speed
- H_m: the set of resources of mode m
- λ_{im}: maximum travel time between the resources of mode m and of task i: $\lambda_{im} = max_{m=1}^{M_i}(l_{im} \in H_m)$

- φ_k:represents the available amount of fuel on resource k at any time

- ρ_k:represents the average amount of fuel burnt in η units of time

5.3.2 Bi-level formulation

In this section we describe a time indexed formulation based on the multi-mode resource assignment and task scheduling problem. In the first step, we describe the upper level that consists in scheduling a set of tasks in order to minimize the makespan. In the second step, we describe the lower level that consists in assigning for each task the available and suitable mode m or the combination of resources such that the total cost is minimized.

Decision variables

- F_i: the completion time of task i where $a_i \leq F_i \leq b_i$

- y_i^t: binary variable equal to 1 if task i at the beginning of period $t \in T$ and 0 otherwise

- z_{im}: binary variable equal to 1 if activity i is processed in mode m and 0 otherwise

Upper level formulation

$$\textbf{Upper Level}: \text{Minimize } C_{max} = max_{i=1}^{N} F_i \tag{5.1}$$

$$\text{Subject to } \sum_{t=0}^{T} y_i^t \leq 1, \quad \text{for } i = 1, .., N, \tag{5.2}$$

$$\sum_{t \in T} t(y_j^t - y_i^t) \geq F_i \quad \forall (i,j) \in P \tag{5.3}$$

$$F_i \geq \sum_{t \in T} t y_i^t + \sum_{m \in M_i} z_{im} p_{im} + \sum_{m \in M_i} z_{im} \lambda_{im}, \quad \forall i \in N_1, \tag{5.4}$$

$$F_i \geq \sum_{t \in T} t y_i^t + d_i + \sum_{m \in M_i} z_{im} \lambda_{im}, \quad \forall i \in N_2, \tag{5.5}$$

$$\sum_{m \in M_i} z_{im} = \sum_{t \in T} y_i^t, \quad \text{for } i = 1, .., N, \tag{5.6}$$

$$y_i^t \quad \text{binary}, a_i \leq F_i \leq b_i \quad \text{for} \quad i = 1, .., N, \quad \forall t \in T. \tag{5.7}$$

The objective function (5.1) minimizes the maximum completion time. Constraints (5.2) impose that each task is started at most at one time over the planning horizon T. Constraints (5.3) ensure that the starting time of a task j that follows task i is greater or equal than the completion of task i. Constraints (5.4) establish that the completion time of task $i \in N_1$ is greater or equal than the sum of the starting time, the processing time and the travel time. Constraints (5.5) is the same as Constraints (5.4) but for task $i \in N_2$ having fixed processing time. Constraint (5.6) ensures that each activity i is assigned to exactly one mode $m \in M_i$. Constraint (5.7) states logical binary and nonnegativity restrictions on decision variables.

Lower level formulation

It is based on the idea of ignoring the detailed sequencing constraints and is largely concerned with the assignment of resources to perform tasks through the binary variables z_{im}.

Let C^*_{max} denotes the optimal solution of the upper level. In the lower level, the objective is to minimize the total cost of using the different resources subject to the maximum completion time is less or equal to C^*_{max}.

$$\textbf{Lower Level}\text{: Minimize } \sum_{i=1}^{N} \sum_{m=1}^{M_i} \sum_{k \in H_m} c_k z_{im} \tag{5.8}$$

$$\text{Subject to } \sum_{i=1}^{N} \sum_{m \in M_i} y_i^t z_{im} w_{km} r_{im} \leq E_k, \quad \forall t \in T \quad \forall k \in R \tag{5.9}$$

$$\sum_{i=1}^{N} \sum_{m \in M_i} z_{im} \frac{(2 \sum_{k \in H_m} l_{ik} + p_{im}) \rho_k}{\eta} \leq \varphi_k, \forall k \in R \tag{5.10}$$

$$z_{im} \text{ binary, for } i = 1, .., N, \quad \forall m \in M_i. \tag{5.11}$$

The objective function (5.8) minimizes the total cost of using the different resources. Constraints (5.9) take care of limited quantity of the renewable resource constraints. Constraints (5.10) guarantee the autonomy of the used resources during tasks' execution.

From a historical point of view, the multi-level optimization is closely related to the economic problem of Stackelberg in the field of game theory [23]. The Stackelberg problems possess as a hierarchical structure similar to the bi-level programming problem, although the lower level is an equilibrium rather than an optimization problem. The bi-level method was initially considered by Bracken and McGill for the production and marketing decision making in military field [10][11].

5.4 Solution approach

The proposed solution approach will take advantage of both the local search engines and the globality of genetic algorithm in the search process.

However, the proposed model described by constraints (5-1, 5-10) can become intractable when a large number of tasks is considered. As a result, simultaneous or full space optimization techniques can fail to solve this model for large instances. Therefore, we propose a novel decomposition scheme for the integration of assigning and scheduling that enables the rigorous solution at reasonable computational expense.

5.4.1 Basic concept and solution framework

The MMASP can be decomposed into a tasks sequencing problem and modes assignment problem. The upper level sequencing problem seek to solve the issue of execute sequence of the tasks in an effort to minimize the total project duration. On the other hand, at the lower level resources assignment problem is investigated, and the solution space is explored on the basis of the minimal resource consumption.

Following the above decomposition scheme, a bi-level search framework is proposed

in this paper that consists of two interating levels, each corresponding to a different search landscape. As such, the proposed solution approach employs an Ant Colony Genetic Approach (ACGA) to perform the core search process at the upper level, and a Genetic Local Search Algorithm (GLSA) at the lower level to improve the modes assignment.

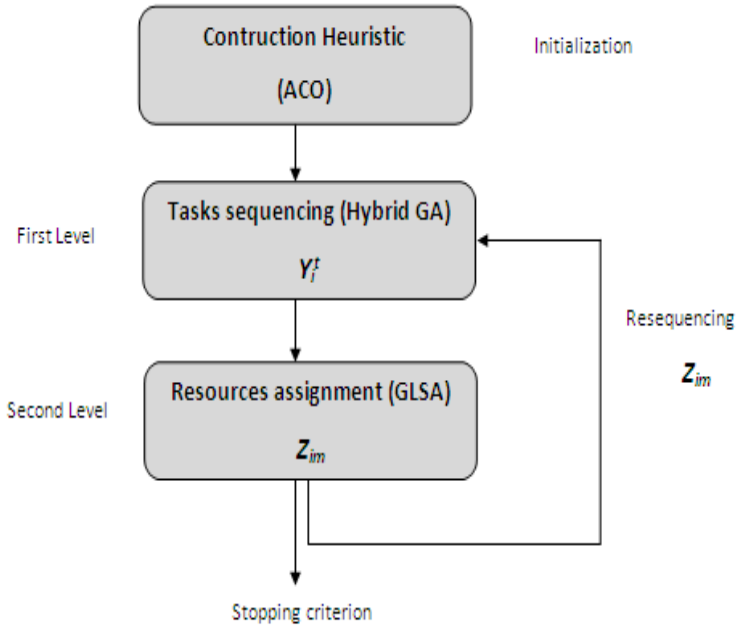

Figure 5.1: Bi-level approach

5.4.2 Construction approach: ACO

To apply the ACO for solving Bi-level ASP, the problem must be reduced to an appropriate graph along which the ant moves to construct candidate solutions. Therefore, we depicted our optimization problem as a disjunctive graph $G = (N, E, D)$, where N is a set of nodes, E is a set of directed edges and D is a set of undirected edges as shown in figure (5.2). The nodes N acted as tasks numbred from 0 to $n + 1$, where the 0th and the $n+1^{th}$ tasks are dummy representing start and end tasks respectively,

E represents the precedence relationships between tasks and D correspond to edge's time F_i and cost c_k. Thus, the problem reduced to search for a minimum cost and time paths in the graph. To manage efficiently both mission cost and time, we used the Bi-Colony Ant Based Approach (BCABA) proposed in [26]. Two types of ants are used to construct a solution. The first type of ant, belonging to the *ACS-cost* colony, is used to optimize the assignment of modes to achieve tasks and the second type, belonging to the *ACS-time* colony, is used to construct efficient schedules. The proposed method builds sequentially the solutions by iteratively distributing the two populations of artificial ant agents on the basis of information provided by both a standard constructive heuristic function and previously constructed solutions via the pheromone trail. Each colony has its own pheromone matrix and each ant of a given colony builds its own solution using only pheromone and heuristic information of its colony.

- ***ACS-cost***

The colony of ant is initially positioned on the first node/task. At each iteration, each ant applies repeatedly a stochastic search function named a state transition rule, to select the appropriate mode to be assigned to the current node/task until it reaches the last node. The state transition rule is directed by both the pheromone value τ_{ik} and the heuristic information η_{ik}.

The heuristic information is a priori value depends on the use cost of resource k, the probability of successfully accomplishing task i A_{ik} and a small constant s. The two first measures are standardized before calculating η_{ik}.

$$\eta_{ik} = (\frac{s}{c_k}) + (1 - \frac{1}{A_{ik}}) \tag{5.12}$$

The heuristic function η_{ik} grows inversely proportional to use cost c_k. The smaller the cost c_k incurred by allocating resource k to task i, the higher the value of η_{ik}.

A pheromone matrix is defined for each colony. The pheromone values reinforce the anterior non dominated solutions. A pheromone matrix $M_I(n \times k)$ is defined for the *ACS-cost* and initialized to τ_0 as follows:

$$\tau_{ik} = \begin{cases} \tau_0 & \text{if resource } k \text{ contributes to the realization of task } i. \\ 0 & \text{otherwise.} \end{cases}$$

In each iteration, the pheromone trails intensity decreases over time to avoid convergence of the algorithm to local optimum and favoring the exploration of not visited areas of the search space according to an evaporation rate ρ, $0 < \rho < 1$.

$$\tau_{ik} = (1 - \rho)\tau_{ik} + \rho\tau_0 \tag{5.13}$$

Once all the ants have completed the tour, if S^c is non-dominated generated solution set according to *ACS-cost* colony, the pheromone values are updated as follows

$$\tau_{ik} = \tau_{ik} + \Delta\tau_{ik} \tag{5.14}$$

With $\Delta\tau_{ik} = \frac{1}{Z_2^*}$, where Z_2^* is the best value according to the second objective function $Z_2 \in S^c$.

Thus, ants choose the adequate mode to be assigned to the current task i using a combination of heuristic and pheromone information. Ant f at node/task i selects the mode m, if $q \leq q_0$:

$$P_{ik} \leftarrow \begin{cases} 1 & if \quad k = argmax_{h \in \aleph_{(i)}} \{\tau_{ih} \times \eta_{ih}\}, \forall\, k \in H_m \\ 0 & otherwise \end{cases}$$

Where q is a random number uniformly distributed in $[0, 1]$, and $0 \leq q_0 \leq 1$ is a predetermined parameter that determines the relative importance of exploitation versus exploration and $\aleph_{(i)}$ denotes the neighborhood set of the resources's combination of the task i. With the state transition rule:

$$P_{ik} \;=\; \frac{(\tau_{ik})^{\alpha}\,(\eta_{ik})^{\beta}}{\sum_{h \in \aleph_{(i)}} \tau_{ih}\;\eta_{ih}} \;\forall\, k \;\in\; \aleph_{(i)} \tag{5.15}$$

With α and β denote the parameters correlated to the importance of the pheromone trail and heuristic information respectively.

- ***ACS-time***

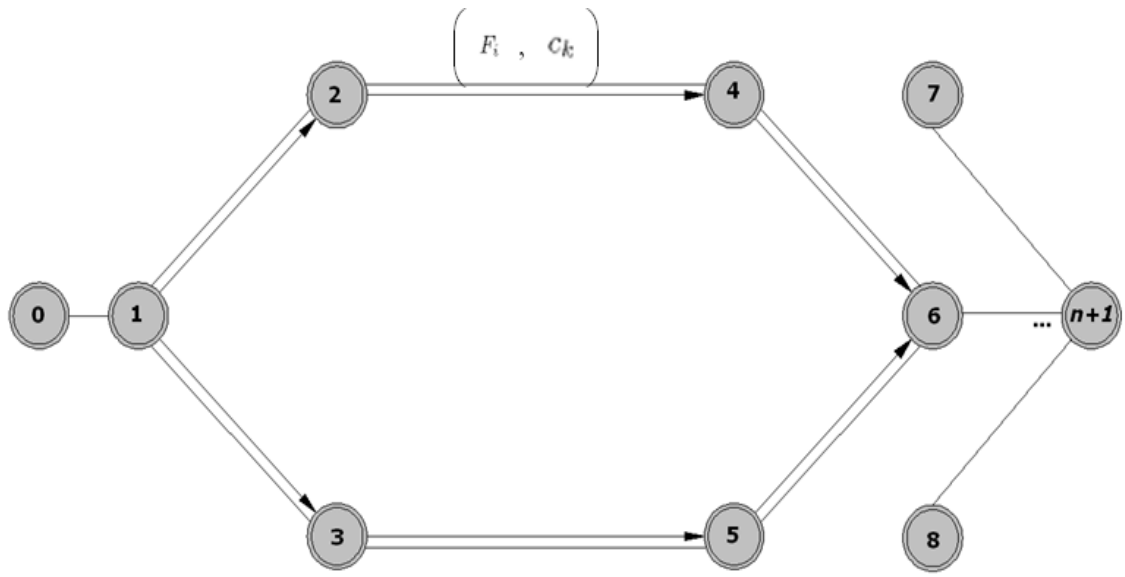

Figure 5.2: Graph representation of an ASP instance

At each iteration one task is selected from the list of non-achieved tasks $L_m(t)$ to be scheduled over a specific period of time $t \in T$ satisfying the problem constraints. Each ant of the colony applies the transition rule $P^t_{iv_m}$ to obtain a neighbor for the current node/task i by the selection of a floating task v to be processed simultaneously with task i at the same period t using any available mode m.
The heuristic function $\eta^t_{iv_m}$ depends on the processing time and the time window constraints of task v.

$$\eta^t_{iv_m} = \frac{1}{(b_v - a_v)} + (1 - \frac{1}{p_{vm}}) \tag{5.16}$$

The pheromone matrix $M_{II}(n \times n)$ for the second colony *ACS-time* is defined as follows

$$\tau_{iv} \leftarrow \begin{cases} \tau_0 & \text{if } v \text{ is a floting task that can be executed simultaneously with } i \\ 0 & \text{otherwise} \end{cases}$$

To diversify the search performed by subsequent ants during an iteration in order to prevent convergence, the pheromone concentration decreases based on equation

(5.17).

$$\tau_{iv} = (1-\rho)\tau_{iv} + \rho\tau_0 \tag{5.17}$$

At the end of the iteration the pheromone values are updated as follows

$$\tau_{iv} = \tau_{iv} + \Delta\tau_{iv} \tag{5.18}$$

With $\Delta\tau_{iv} = \frac{1}{Z_1^*}$, where Z_1^* is the best value according to the first objective function $Z_1 \in S^t$.

If $q \leq q_0$, the ant chooses the task v to be processed simultaneously with task i during period t such as

$$P_{iv_m}^t \leftarrow \begin{cases} 1 & if \quad v = argmax_{u \in \varphi_{(i)}} \{\tau_{iu} \times \eta_{iu}\} \\ 0 & otherwise \end{cases}$$

Where $\varphi_{(i)}$ is the floting task set and $P_{iv_m}^t$ the transition rule:

$$P_{iv_m}^t = \frac{(\tau_{iv})\,(\eta_{iv_m}^t)}{\sum_{u \in \varphi_{(i)}} \tau_{iu}\ \eta_{iu}} \ \forall\, v \in \varphi_{(i)} \tag{5.19}$$

The basic outline of the algorithm is summarized as follows:

Bi-colony Ant based algorithm

$t = 1$
f= number of ants of each colony
Initialize the pheromone matrices M_I and M_{II}
Assign the ants to the starting task
$S^c \leftarrow \emptyset$
$S^t \leftarrow \emptyset$
$S \leftarrow S^c \cup S^t$
While stopping criteria not met **Do**
$g \leftarrow 1$
Do
Activate *ACS-cost*(M_I)
Activate *ACS-time*(M_{II})
Perfom a local update according to Eqs. (5.13) and (5.17)
Update the non dominated list S^c
Update the non dominated list S^t
$g \leftarrow g + 1$
While $g \leq f$
Perform global update based on Eqs. (5.14) and (5.18)
$S \leftarrow S^c \cup S^t$
Filter the list S
End While

5.4.3 Upper level search strategy - C_{max} minimization

This section presents the upper level search strategies that aim to minimize the project makespan. The sequencing phase, based on an ACGA, is intended as an intensification phase that optimize the tasks sequence without changing the set of assigned/used

modes.
The basic steps of the algorithm are as follows:

- Step 1: Create the initial population of S individuals. Usually in GA the initial population is randomly generated which may or may not contain good candidate solutions. To avoid this latter case, and to guarantee that method will provide feasible solution chromosomes, an ACO algorithm is used to generate the initial population to be improved and evolved during the genetic process.

- Evaluate the fitness of each individual
- Select parents from the current population and choose via roulette wheel selection the pairs for mating
- Apply a random key crossover operator according to $cross_p$
- Improve some offspring by mutation operator

5.4.4 Lower level search strategy - Cost minimization

5.5 Empirical results

5.6 Conclusion

Table 5.1: Computational results

Instance	***#Activities***	***#Resources***	$\boldsymbol{Avg_{pred}}$	***CPU(s)***	$\boldsymbol{Gap_{LB}}$	$\boldsymbol{Cov_Z}\%$
I_1	6	3	$(1-2)$	0.07	0	80
I_2	6	4	$(1-2)$	0.07	0	80
I_3	6	5	$(1-2)$	0.07	0	90
I_4	6	5	$(2-3)$	0.07	0.05	27
I_5	11	12	$(1-3)$	0.1	0.03	47
I_6	30	4	$(0-2)$	0.2	0.03	58
I_7	30	4	$(0-4)$	0.2	0.01	43
I_8	50	3	$(1-2)$	0.29	0.01	1
I_9	50	3	$(2-3)$	0.31	0.011	35
I_{10}	50	10	$(1-2)$	0.35	0.02	50
I_{11}	60	4	$(1-2)$	0.35	0.03	45
I_{12}	100	3	$(1-2)$	0.61	0.02	22
I_{13}	100	3	$(2-3)$	0.61	0.18	26
I_{14}	100	5	$(2-3)$	0.66	0.22	12
I_{15}	100	15	$(1-2)$	0.69	0.08	27
I_{16}	150	15	$(1-2)$	1.07	0.08	32
I_{17}	200	20	$(1-2)$	1.51	0.07	32
			Avg	0.42	0.06	41

General conclusion

Bibliography

Bibliography

[1] Agin, N. [1966]: "Optimum seeking with Branch and Bound", Management Science, Vol. 13, No. 4; pp. 176-185.

[2] Aissi, H., Bazgan, C. and Vanderpooten, D. [2008]: "Min-max and min-max regret versions of combinatorial optimization problems: A survey", to appear in European Journal of Operational Research.

[3] Alcaraz, J. and Maroto, C. [2001]: "A Robust Genetic Algorithm for Resource Allocation in Project Scheduling",Annals of Operations Research, Vol. 102; pp. 83-109.

[4] Alcaraz, J., Maroto, C. and Ruiz, R. [2003]: "Solving the Multi-Mode Resource-Constrained Project Scheduling Problem with genetic algorithms", Journal of the Operational Research Society, Vol.54; pp. 614-626.

[5] Badia, L., Botta, A. and Lenzini, L. [2009]: "A genetic approach to joint routing and link scheduling for wireless mesh networksl", Ad Hoc Networks, Vol.7; pp. 654-664.

[6] Badiru, A.B. [1993]: "Quantitative Models for Project Planning, Scheduling, and Control", Quorum Books, Connecticut in United States of America.

[7] Belfares, L., Klibi, W., Lo, N. and Guitouni, A. [2007]: "Multi-objectives Tabu Search based algorithm for progressive resource allocation", European Journal of Operational Research, Vol. 177; pp. 1779-1799.

[8] Ben Abdelaziz, F., Krichen, S. and Dridi, O. [2007]: "A Multiobjective Resource-Constrained Project-Scheduling Problem", Lecture Notes in Computer Science (LNAI), Springer Verlag, Vol.4724; pp. 719-730.

[9] Blazewiez, J., Lenstra, J.K. and Rinnooy Kan, A.H.G. [1983]: "Scheduling subject to resource consytraints: Classification and Comlexity", Discrete Applied Mathematics, Vol.5; pp. 11-24.

[10] Bracken, J. and McGill, J. [1973]: "Mathematical programs with optimization problems in the constraints", Operations Research, Vol.21; pp. 37-44.

[11] Bracken, J. and McGill, J. [1974]: "Defense applications of mathematical programs with optimization problems in the constraints", Operations Research, Vol.22, No.5; pp. 1086-1096.

[12] Bretthauer, K.M. and Shetty, B. [1995]: "The nonlinear resource allocation problem", Operational Reserach, Vol.43; pp. 670-683.

[13] Bretthauer, K.M. and Shetty, B. [2002]: "A pegging algorithm for the nonlinear resource allocation problem", Computer of Operational Reserach, Vol.29; pp. 505-527.

[14] Brucker, P., Drexl, A., Möhring, R., Neumann,K. and Pesch, E. [1999]: "Resource-constrained project scheduling: Notation, classification, models, and methods", European Journal of Operational Research, Vol.112; pp 3-41.

[15] Chaharsooghi, S.K. and Kermani, A.H.M. [2008]: "An effective ant colony optimization algorithm (ACO) for multi-objective resource allocation problem (MORAP)", Applied Mathematics and Computation, Vol.200; pp. 167-177.

[16] Cicirello, V.A. and Smith, S.F. [2001]: "Ant colony control for autonomous decentralized shop floor routing", IEEE proceedings of the Fifth International Symposium on Autonomous Decentralized systems; pp 383-390.

[17] Dai, Y.S., Xie, M., Poh, K.L. and Yang, B. [2003]: "Optimal testingresource allocation with genetic algorithm for modular software systems", Journal of Systems and Software, Vol.66; pp. 47-55.

[18] Dantzig, G.B. [1963]: "Linear Programming and Extensions", Princeton University Press.

[19] Dammak, A., Elloumi, A., Kamoun, H. [2006]: "Classroom assignment for exam timetabling", Advances in Engineering Software, vol.37; pp 659-666.

[20] De, P., Dunne, E.J., Gosh, J.B. and Wells, C.E. [1997]: "Complexity of the Discrete Time-Cost Tradeoff Problem for Project Networks, Operations Research", Vol. 45; pp. 302-306.

[21] Debels, D. and Vanhoucke, M. [2005]: "A Bi-population Based Genetic Algorithm for the Resource-Constrained Project Scheduling Problem", Lecture Notes on Computer Science, Vol.3483; pp. 378-387.

[22] Demeulemeester, E. and Herroelen, W. [1992]:" A Branch and Bound procedure for the Multiple Resource-Constrainted-project-Scheduling Problem", Management Science, Vol.38, No.12; pp. 1803-1818.

[23] Dempe, S. [2003]: "Annotated bibliography on bilevel programming and mathematical programs with equilibrium constraints", Optimization, Vol. 52; pp. 333-359.

[24] Dorigo, M. (1992): Optimization, learning and natural algorithms. *PhD thesis*, Politecnico di Milano, Italy.

[25] Dowsland Kathryn, A. [1996]: "Genetic Algorithms-A tool for OR? ", The Journal of the Operational Research Society, Vol.47, No.4; pp.550-561.

[26] **Dridi, O., Krichen, S. and Guitouni, A. (2012): A Multi-Objective Optimization Approach for Resource Assignment and Task Scheduling Problem: Application to Maritime Domain Awareness.** *IEEE Congress on Evolutionary Computational (CEC'12)*, **pp.1-8.**

[27] Ebben, M.J.R., Van Der Heijden, M.C. and Van Harten, A. [2005]: "Dynamic transport scheduling under multiple resources constraints", European Journal of Operational Research, Vol.167; pp 320-335.

[28] Elbeltagi, E., Hegazy, T. and Grierson, D. [2005]: "Comparison among five evolutionary-based optimization algorithms ", Advanced Engineering Informatics; pp.1-11.

[29] Erdirik-Dogan, M., and Grossmann, I. E. [2006]: "A decomposition method for the simultaneous planning and scheduling of single-stage continuous multiproduct plants", Industrial Engineering and Chemical Research, Vol.45; pp. 299-315.

[30] Erdirik-Dogan, M., and Grossmann, I. E. [2008]: "Simultaneous planning and scheduling of single-stage multi-product continuous plants with parallel lines. Computers and Chemical Engineering", Vol. 32; pp. 26642683.

[31] L. J. Fogel, A. J. Owens, and M. J. Walsh. Artificial Intelligence Through Simulated Evolution. Wiley, 1966.

[32] Gen.M, Lin.L and Zhang.H [2008]: "Evolutionary techniques for optimization problems in integrated manufacturing system: State-of-the-art-survey", to appear in Computers and Industrial Engineering.

[33] Goldberg.D.E [1989]: "Genetic Algorithms in Search, Optimization and Machine Learning", Addison-Wesley Publishing Company.

[34] Gonçalves.J.F, Mendes.J.J.M and Resende.M.G.C [2008]: "A genetic algorithm for the resource constrained multi-project scheduling problem", European Journal of Operational Research, Vol.189; pp. 1171-1190.

[35] Graham.R, Lawler. E, Lenstra.J and Rinnooy Kan.A [1979]: "Optimization and approximation in deterministic sequencing and scheduling: A survey", Annals of Discrete Mathematics, Vol.5; 287-326.

[36] Grob, M. [2006]: "Routing of platforms in a maritime surface surveillance operation", European Journal of Operational Research, Vol. 170; pp. 613628.

[37] Grünwald.P.D and Dawid.A.P [2004]: "Game Theory, Maximun Entropy, Minimum Discrepancy and Robust Bayesian Decision Theory", The Annals of Statistics, Vol.32, No.4; pp. 1367-1433.

[38] Guo, Z.X, Wong, W.K., Leung, S.Y.S., Fan, J.T., and Chan, S.F. [2006]: "A Bi-level Genetic Algorithm for Multi-objective Scheduling of Multi- and Mixed-Model Apparel Assembly Lines", Sattar, A., and Kang, B.H. (Eds.): AI, LNAI, 4304;pp. 934-941.

[39] Hartmann.S and Sprecher.A [1996]: "A note on "Hierarchical models for multi-project planning and scheduling", European Journal of Operational Research, Vol.94; pp 377-383.

[40] Heilmann.R [2003]: "A branch-and-bound procedure for the multi-mode resource-constrained project scheduling problem with minimum and maximum time lags", European Journal of Operational Research, Vol.144; pp. 348-365.

[41] Herroelen.W and Leus.R [2005]: "Project scheduling under uncertainty: Survey and research potentials", European Journal of Operational Research, Vol.165; pp. 289-306.

[42] Herroelen.W Reyck.B.D and Demeulemeester.E [1998]: "Resource-constrained project scheduling: A Survey of recent developments", Computer of Operational Research, Vol.25, No.4; pp. 279-302.

[43] Herzog.F, Dondi.G, Keel.S, Schuman.L.M and Geering.H.P [2007]: "Solving ALM problems via sequential stochastic programming", Quantitative Finance, Vol.7, No.2; pp. 231-244.

[44] Hicks.C, Song.D.P and Earl.C.F [2007]: "Dynamic scheduling for complex engineer-to-order products", International Journal of Production Research, Vol.45, No.15; pp. 3477-3503.

[45] Hindelang, T.J. and Muth, J.F. [1979]: "A Dynamic Programming Algorithm for Decision CPM Networks", Operations Research, Vol. 27; pp. 225-241.

[46] Ho.S.C and Leung.J.M.Y [2010]: "Solving a manpower scheduling problem for airline catering using metaheuristics", European Journal of Operational Research, Vol.202; pp. 903-921.

[47] Hoogeveen.H [2005]: "Multicriteria Scheduling", European Journal of Operational Research, Vol.167; pp. 592-623.

[48] J. H. Holland. Adaptation in Natural and Artificial Systems.University ofMichigan Press, Ann Arbor, MI, 1975.

[49] Horn.M, Jiang.H and Kilby.P [2007]: "Scheduling patrol boats and crews for the Royal Australian Navy", Journal of the Operational Research Society, Vol. 58;pp. 1284-1293.

[50] Jackson J.R [1955]: "Scheduling a production line to minimize maximum lateness", Research Report, Management science research report, University of California, Los Angeles.

[51] Jeeva.M, Rajalakshmi Rajagopal, Charles.V and Yadavalli.V.S.S [2004]: "An Application of Stochastic Programming with Weibull DistributionCluster Based Optimum Allocation of Recruitment in Manpower Planning", Stochastic Analysis and Applications, Vol.22, No.3; pp. 801-812.

[52] Jennings.N.R [2000]: "On agent-based software enginnering", Artificial Intelligence, Vol.117, No.2; pp. 277-296.

[53] Johnson.S [1954]: "Optimal two- and three-stage production schedules with setup times included", Naval Research Logistics Quarterly Vol.1; pp. 6168.

[54] Karlof, J.K., and Wang, W. [1996]: "Bi-level Programming Applied to the Flow Shop Scheduling Problem", Computers of Operationals Research, Vol.23, No.5; pp. 443-451.

[55] Karp.R.M [1972]: "Reducibility among combinatorial problems", Complexity of Computer Computations, Eds. R.E.Miller and J.W. Thatcher, Plenum Press; pp. 85103.

[56] Kelly, J.E. [1961]: "Critical path planning and scheduling: mathematical basis, Operational Research, Vol.9, No.3; pp. 296-320.

[57] Kennedy, J. and Eberhart, R. C. (1995): Particle swarm optimization. *In IEEE International Conference on Neural Networks*, pp. 19421948.

[58] kolisch.R and Hartmann.S [2000]: "Experimental evaluation of state-of-the-art heuristics for the resource-constrained project Scheduling problem", European Journal of Operational Research Vol.127; pp.394-407.

[59] kolisch.R and Hartmann.S [2006]: "Experimental investigation of heuristics for resource-constrained project scheduling: An update", European Journal of Operational Research Vol.174; pp. 23-37.

[60] Korhonen.P and Syrjänen.M [2004]: "Resource Allocation Based on Efficiency Analysis", Management Science, Vol.50, No.8; pp. 1134-1144.

[61] J. R. Koza. Genetic Programming. MIT Press, Cambridge, MA, 1992

[62] Kreucher, C., Blatt, D., Hero, A. and Kastella, K., [2006]: "Adaptive multimodality sensor scheduling for detection and tracking of smart targets", Digital Signal Processing, Vol. 16, pp. 546-567.

[63] Lasserre, J.B. [1992]: "An intergrated model for job-shop planning and scheduling", Management Science, Vol. 38,pp. 1201-1211.

[64] Lee.Z.J and Lee.C.Y [2005]:"A hybrid search algorithm with heuristics for resource allocation problem", Information Sciences Vol.173; pp. 155-167.

[65] Li.Y.P, Huang.G.H, Nie.S.L and Huang.Y.F [2006]: "IFTSIP: interval fuzzy two-stage stochastic mixed-integer linear programming: a case study for environmental management and planning", Civil Engineering and Environmental Systems, Vol.23, No.2; pp. 73-99.

[66] Li.Y, Krakow.L.W, Chong.E.K.P and Groom.K.N [2008]: "Approximate stochastic dynamic programming for sensor scheduling to track multiple targets", to appear in Digital Signal Processing.

[67] Lin.C.M and Gen.M [2008]:"Multi-criteria human resource allocation for solving multistage combinatorial optimization problems using multiobjective hybrid genetic algorithm", Expert Systems with Applications, Vol.34; pp. 2480-2490.

[68] Lin, F.R., Shaw, M.J. and Locascio, A. [1997]: "Scheduling Printed Circuit Board Production Systems Using the Two-Level Scheduling Approach", Journal of Manufacturing Systems, Vol.16, No. 2, pp. 129-149.

[69] Liu.H, Wan.P and Jia.X [2006]: "Maximal lifetime scheduling for K to 1 sensor-target surveillance networks", Computer Networks, Vol.50; pp. 2839-2854.

[70] Lo.S.T, Chen.R.M, Huang.Y.M and Wu.C.L[2008]: "Multiprocessor system scheduling with precedence and resource constraints using an enhanced ant colony system", Expert Systems with Applications, Vol. 34; pp 2071-2081.

[71] Loukil.T, Teghem.J and Tuyttensb.D [2005]: "Solving multi-objective production scheduling problems using metaheuristics", European Journal of Operational Research, Vol.161; pp. 42-61 .

[72] Lova.A, Tormos.P, Cervantes.M and Barber.F [2009]: "An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes", International Journal of Production Economics, Vol.117; pp. 302-316.

[73] Low.K.H, Leow.W.K and M.H.Ang.J [2004]: "Task Allocation via Self-Organization Swarm Coalitions in Distributed Mobile Sensor Netwoek", In proceedings of the American Association of Artificial Intelligente; pp. 28-33.

[74] Lu.G and Krishnamachari.B [2007]: "Minimum latency joint scheduling and routing in wireless sensor networks", Ad Hoc Networks, Vol.5; pp. 832-843.

[75] Mahmoud.M.R [2006]: "High dimension dynamic programming model for water resources expansion projects", Engineering Optimization, Vol.38, No.3; pp. 371-389.

[76] Mautor.T and Roucairol.C [1994]:"A new exact algorithm for the solution of quadratic assignment problems", Discrete Applied Mathematics, Vol.55; pp. 281-293.

[77] Mendes.J.J.M, Gonçalves.J.F and Resende.M.G.C [2009]: "A random key based genetic algorithm for the resource constrained project scheduling problem", Computer and Operational Research, Vol.36; pp. 92-109.

[78] Meyer, W.L. and Shaffer, L.R. [1963]: "Extensions of the critical path method through the application of integer programming, Civil Engineering Construction Research Series, No.2, University of Illinois, Urbana-Champaign.

[79] Mingozzi.A, Maniezzo.V, Ricciardelli.S and Bianco.L [1998]: "An exact algorithm for the multiple resource-constrained project scheduling problem based on a new mathematical formulation", Management Science, Vol.44, No.5; pp. 714729.

[80] Ng.K and Sancho.N [2009]: "Regional surveillance of disjoint rectangles: a travelling salesman formulation", Journal of the Operational Research Society, Vol.60; pp. 215-220.

[81] Overholts.L.D, Bell.J.E and Arostegui.M.A [2009]: "A location analysis approach for military maintenance scheduling with geographically dispersed service areas", Omega, Vol.37; pp. 838-852.

[82] Peteghem.V.V and Vanhoucke.M [2010]: "A genetic algorithm for the preemptive and non-preemptive multi-mode resource-constrained project scheduling problem", European Journal of Operational Research, Vol.201; pp. 409-418.

[83] Pemberton.J.C and Greenwald.L.G [2002]: "On the Need for Dynamic Scheduling of Imaging Satellites", Pecora 15/Land Satellite Information IV/ISPRS Commission I/FIEOS Conferance Proceedings.

[84] Piantadosi.J, Metcalfe.A.V and Howlett.P.G [2008]: "Stochastic dynamic programming (SDP) with a conditional value-at-risk (CVaR) criterion for management of storm-water", Journal of Hydrology, Vol.348; pp. 320-329.

[85] Pritsker.A.A.B, Watters.L.J and Wolfe.P.M [1969]: "Multi-project scheduling with limited resources: A zero-one programming approach, Management Science, Vol.16; pp. 93-108.

[86] Ranjbar, M., Khalilzadeh, M., Kianfar, F. and Etminani, K. [2011]: "An optimal procedure for minimizing total weighted resource tardiness penalty costs in the resource-constrained project scheduling problem", Computers & Industrial Engineering, in press.

[87] Rechenberg I. (1994). Evolutionary strategy. In Computational Intelligence-mitating Life, pp 147–159. IEEE press, Piscataway, USA.

[88] Robinson, D.R. [1975]: "A Dynamic Programming Solution to Cost-Time Trade-Off for CPM", Management Science, Vol. 22; pp 158-166.

[89] Shen, W.M. (2002): Distributed manufacturing scheduling using intelligent agent. *IEEE Intelligent Systems*, Vol.17, No.1; pp. 88-94.

[90] Shen.W., Maturana.F, Norrie.D and Barthès.J.P (1997): Agent-Based Approaches For Advanced CAD/CAM Systems. *In Proceedings of the Fifth International Conference on CAD/Graphics*, Shenzhen, China; pp. 609-615.

[91] Shen.W. and Norrie.D (1998): "An Agent-Based Approach for Dynamic Manufacturing Scheduling", Proceedings of the 3rd International Conference on the Practical Applications of Agents and Multi-Agent Systems (PAAM-98).

[92] Silvaa.C.A, Sousaa.J.M.C, Runklerb.T.A (2008): "Rescheduling and optimization of logistic processes using GA and ACO", to appear in Engineering Applications of Artificial Intelligence.

[93] Simpson.W.P and Patterson.J.H (1996): "A multiple-tree search procedure for the resource-constrained project scheduling problem", European Journal of Operational Research, Vol.89; pp. 525-542.

[94] Slowinski.R [1980]: "Two Approaches to Problems of Resource Allocation Among Project Activities- A Comparative Study", Journal of Operational Research Society, Vol.31; pp. 711-723.

[95] Smith, W.E. [1956]: Various Optimizers for Single Stage Production. *Naval Research Logistics Quarterly*, Vol. 3; pp. 5966.

[96] Speranza, M., and Vercellis, C. [1993]: Hierarchical models for multi-project planning and scheduling. *European Journal of Operational Research*, Vol.64, No.2; pp. 312-325.

[97] Sprecher. A., Hartmann. S., and Drexl. A. [1997]: An exact algorithm for project scheduling with multiple modes, *OR Spektrum*, Vol.19; pp. 195-203.

[98] Suwa.H and Fujiwara.T [2007]: "A New Rescheduling Policy Based On Cumulative Delay", 19th International Conference on Production Research.

[99] Talbot.F.B and Patterson.J.H [1978]: "An efficient integer programming algorithm with network cuts for solving resource-constrained scheduling problems", Management Science, Vol.24, No.11; pp. 1163-1174.

[100] Tanga.L, Xuana.H and Liub.J [2007]: "Hybrid backward and forward dynamic programming based Lagrangian relaxation for single machine scheduling", Computers and Operations Research, Vol.34; pp 2625-2636.

[101] Thomas.M.E [1976]: "A Survey of the State of the Art in Dynamic Programming ", IIE Transactions, vol.8; pp. 59-69.

[102] Trojet, M., H'Mida, F. and Lopez, P. [2011]: "Project scheduling under resource constraints: Application of the cumulative global constraint in a decision support framework", Computers & Industrial Engineering, Vol.61; pp. 357-363.

[103] Vieira.G.E, Herrmann.J.W and Lin.E [2003]: "Rescheduling manufacturing systems: a framework of strategies, policies and methods", Journal of Scheduling, Vol.6, No.1; pp. 36-92.

[104] Vonder.S.V, Demeulemeester.E and Herroelen.W [2007]: "A classification of predictive-reactive project scheduling procedures", Journal of Scheduling, Vol.10; pp. 195-207.

[105] Wanh.S-J, Xi.L-F, Zhou.B-H [2007]: "Filtered-beam-search-based algorithm for dynamic rescheduling in FMS", Robotics and Computer-Integrated Manufacturing, Vol.23; pp 457-468.

[106] Wu, J. and Azarm, S., [2001]: "Metrics for quality assessment of a multiobjective design optimization solution set", Transaction of the ASME, Vol.123; pp. 18-25.

[107] Xiang.W and Lee.H.P [2008]: "Ant colony intelligence in multi-agent dynamic manufacturing scheduling", Engineering Applications of Artificial Intelligence, Vol.21; pp. 73-85.

[108] Yang, B., Geunes, J. and O'Brien, W.J., [2001]: "Resource-Constrained Project Scheduling: Past Work and New Directions", Research report, Department of Industrial and Systems Engineering, University of Florida.

[109] Yang.B and Geunes.J [2008]: "Predictive-reactive scheduling on a single resource with uncertain future jobs", European Journal of Operational Research, Vol.189; pp. 1267-1283.

[110] Zamani, R., [2004]: "An efficient time-windowing procedure for scheduling projects under multiple resource constraints", OR Spectrum, Vol.26; pp. 423-440.

[111] Zheng, Y., Wana, Z. and Wang, G., [2011]: "A fuzzy interactive method for a class of bilevel multiobjective programming problem", Expert Systems with Applications, Vol. 38; pp. 10384-10388.

[112] Zhu, G., Bard, F.J. and Tu, G., [2006]:"A Branch-and-Cut Procedure for the Multimode Resource-Constrained Project-Scheduling Problem", Journal on Computing, Vol.18, No.3; pp. 377-390.

[113] J. Alcaraz and C. Maroto. "A Robust Genetic Algorithm for Resource Allocation in Project Scheduling". *Annals of Operations Research*, vol. 102, pp. 83–109, 2001.

[114] J. Alcaraz, C. Maroto and R. Ruiz. "Solving the multi-mode resource-constrained project scheduling problem with genetic algorithms". *Journal of Operation Research Society* 54, 614–626, 2003.

[115] J. Bean. "Genetic algorithms and random keys for sequencing and optimization". *ORSA Journal on Computing*, vol. 6, pp. 154-160, 1994.

[116] F. Ben Abdelaziz. "An Interactive Method for the Biobjective Resource-Constrained Project Scheduling". to appear in *International Journal of Multicriteria Decision Making*.

[117] P.A.N. Bosman and D. Thierens. "The balance between proximity and diversity in multi-objectives evolutionary algorithms". *IEEE Tans. Evol. Comput.*, vol. 7(2), pp. 174-188, 2003.

[118] C.M. Fonseca and P.J. Fleming. "Genetic algorithm for multi-objective optimization: Formulation, discussion and generalization". *International Conferance on Genetic Algorithms (ICGA'93)*, pp. 416–423, 1993.

[119] M.P. Fourman. "Compaction of symbolic layout using genetic algorithm". *International Conference on Genetic Algorithms*, pp. 141-153, 1985.

[120] C.A. Coello Coello, D.A. Van Veldhuizen and G.B. Lamont. "Evolutionary Algorithms for Solving Multi-objective Problems". New York, USA: Kluwer Academic Publishers, 2002.

[121] N. Damak, B. Jarboui, P. Siarry and T. Loukil. "Differential evolution for solving multi-mode resource-constrained project scheduling problems2009. *Computers and Operations Research*, vol. 36, pp. 2653–2659, 2009.

[122] M. Ehrgott and X. Gandibleux. "Multiple Cretiria Optimization: State of the Art Annotated Bibliographic Surveys". *Kluwer Academic Publishers*, 2003.

[123] D.E. Goldberg. "Genetic Algorithms in Search, Optimization and Machine Learning". *Addison-Wesley Publishing Company*, 1989.

[124] P.D. Grünwald and A.P. Dawid. "Game Theory, Maximun Entropy, Minimum Discrepancy and Robust Bayesian Decision Theory". *The Annals of Statistics*, vol. 32(4), pp. 1367–1433, 2004.

[125] S. Hartmann. "Project Scheduling with Multiple Modes: A Genetic Algorithm". *Annals of Operations Research*, vol. 102, pp. 111–135, 2001.

[126] H. Hoogeveen. "Multicriteria Scheduling". *European Journal of Operational Research*, vol.167; pp. 592-623, 2005.

[127] Z.J. Lee and C.Y. Lee. "A hybrid search algorithm with heuristics for resource allocation problem". *Information Sciences*, vol. 173, pp. 155–167, 2005.

[128] J.J.M. Mendes, J.F. Gonçalves and M.G.C. Resende. "A random key based genetic algorithm for the resource constrained project scheduling problem". *Computer and Operational Research*, vol. 36, pp. 92–109, 2009.

[129] M. Mori and C.C. Tseng. "A genetic algorithm for multi-mode resource-constrained project schduling problem". *European Journal of Operational Research*, vol. 100, pp. 134–141, 1997.

[130] A. Lova, P. Tormos, M. Cervantes and F. Barber. "An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes". *International Journal of Production Economics*, vol. 117(2), pp. 302–316, 2009.

[131] J.D. Schaffer. "Multiple objective optimization with vector evaluated genetic algorithm". International Conferance on Genetic Algorithms (ICGA'85), pp. 93–100, 1985.

[132] N. Srinivas and K. Deb. "Multi-objective optimization using nondominated sorting in genetic algorithms". *Evolutionary Computation*, vol. 2(3), pp. 221–248, 1994.

[133] J. Wu and S. Azarm. "Metrics for quality assessment of a multiobjective design optimization solution set". *Transaction of the ASME*, vol. 123, pp. 18–25, 2001

[134] E. Zitzler and L. Thiele. "Multi-objective evolutionary algorithms: A comparative case study and the strength Pareto approach". *IEEE Trabsactions on Evolutionary Computation*, vol. 3(4), pp. 257–271, 1999.

[135] E. Zitzler, M. Laumanns, C.M. Fonseca and V.G. da Fonseca. "Performance assessment of multiobjective optimizers: an analysis and review". *IEEE Tans. Evol. Comput.*, vol. 7(2), pp. 117-132, 2003.

[136] A. Agarwal, S. Colak and S. Erenguc. "A neurogenetic approach for the resource constrained project scheduling problem". *Computers & Operations Research*, vol. 38, pp. 44–50, 2011.

[137] J. Alcaraz and C. Maroto. "A Robust Genetic Algorithm for Resource Allocation in Project Scheduling". *Annals of Operations Research*, vol. 102, pp. 83–109, 2001.

[138] J. Alcaraz, C. Maroto and R. Ruiz. "Improving the performance of genetic algorithms for the RCPS problem". *Proceedings of the 9th International Workshop on Project Management and Scheduling*, pp. 40–, 2004.

[139] J. Bean. "Genetic algorithms and random keys for sequencing and optimization". *ORSA Journal on Computing*, vol. 6, pp. 154-160, 1994.

[140] L. Belfares, W. Klibi, N. Lo and A. Guitouni. "Multi-objectives Tabu Search based algorithm for progressive resource allocation". *European Journal of Operational Research*, vol. 177, pp. 1779–1799, 2007.

[141] J. Blazewiez, J.K. Lenstra and A.H.G. Rinnooy Kan. "Scheduling subject to resource constraints: Classification and Complexity". *Discrete Applied Mathematics*, vol. 5, pp. 11–24, 1983.

[142] 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*, vol. 149, pp. 268–281, 2003.

[143] P. Brucker, A. Drexl, R. Möhring, K. Neumann and E. Pesch. "Resource-constrained project scheduling: Notation, classification, models, and methods". *European Journal of Operational Research*, vol. 112, pp. 3–41, 1999.

[144] R.M. Chen. "Particle swarm optimization with justification and designed mechanisms for resource-constrained project scheduling problem", *Expert Systems with Applications*, vol. 38, pp. 7102–7111,2011.

[145] W. Chen, Y.J. Shi, H.F. Teng, X.P. Lan and L.C. Hu. "An efficient hybrid algorithm for resource-constrained project scheduling". *Information Sciences*, vol. 180, pp. 1031–1039, 2010.

[146] S.K. Chaharsooghi and A.H.M. Kermani. "An effective ant colony optimization algorithm (ACO) for multi-objective resource allocation problem (MORAP)". *Applied Mathematics and Computation*, vol. 200, pp. 167–177, 2008.

[147] A. Dammak, A. Elloumi and H. Kamoun. "Classroom assignment for exam timetabling". *Advances in Engineering Software*, vol. 37, pp. 659–666, 2006.

[148] D. Debels, B. De Reyck, R. Leus and M. Vanhoucke. "A hybrid scatter search/electromagnetism meta-heuristic for project scheduling". *European Journal of Operational Research*, vol. 169, pp. 638–53, 2006.

[149] E. Demeulemeester and W. Herroelen. "A Branch and Bound procedure for the Multiple Resource-Constrainted-project-Scheduling Problem". *Management Science*, vol. 38, pp. 1803-1818, 1992.

[150] G.F. Deng and W.T. Lin. "Ant colony optimization-based algorithm for airline crew scheduling problem". *Expert Systems with Applications*, vol. 38, pp. 5787–5793, 2011.

[151] C. Fang and L.Wang. "An effective shuffled frog-leaping algorithm for resource-constrained project scheduling problem".*Computers & Operations Research*, vol. 39, pp. 890–901, 2012.

[152] S. Hartmann. "A competitive genetic algorithm for resource-constrained project scheduling". *Naval Research Logistics*, vol. 45, pp 279–302, 1998.

[153] D.E. Goldberg. "Genetic Algorithms in Search, Optimization and Machine Learning". *Addison-Wesley Publishing Company*, 1989.

[154] W. Herroelen, B.D. Reyck and E. Demeulemeester. "Resource-constrained project scheduling: A Survey of recent developments". *Computer of Operational Research*, vol. 25(4), pp. 279–302, 1998.

[155] R. kolisch and S. Hartmann. "Experimental evaluation of state-of-the-art heuristics for the resource-constrained project Scheduling problem". *European Journal of Operational Research*, vol. 127, pp. 394–407, 2000.

[156] R. kolisch and S. Hartmann. "Experimental investigation of heuristics for resource-constrained project scheduling: An update". *European Journal of Operational Research*, vol. 174, pp. 23–37, 2006.

[157] R. kolisch and A. Sprecher. "PSPLIBA project scheduling problem library". *European Journal of Operational Research*, vol. 96, pp. 205–216, 1996.

[158] Z.J. Lee and C.Y. Lee. "A hybrid search algorithm with heuristics for resource allocation problem". *Information Sciences*, vol. 173, pp. 155–167, 2005.

[159] H. Liu, P. Wan and X. Jia. "Maximal lifetime scheduling for K to 1 sensor-target surveillance networks". *Computer Networks*, vol. 50, pp. 2839–2854, 2006.

[160] S.T. Lo, R.M. Chen, Y.M. Huang and C.L. Wu. "Multiprocessor system scheduling with precedence and resource constraints using an enhanced ant colony system". *Expert Systems with Applications*, vol. 34, pp. 2071-2081, 2008.

[161] T. Loukil, J. Teghem and D. Tuyttens. "Solving multi-objective production scheduling problems using metaheuristics". *European Journal of Operational Research*, vol. 161, pp. 42–61, 2005.

[162] J.J.M. Mendes, J.F. Gonçalves and M.G.C. Resende. "A random key based genetic algorithm for the resource constrained project scheduling problem". *Computer and Operational Research*, vol. 36, pp. 92–109, 2009.

[163] K. Nonobe and T. Ibaraki. "Formulation and tabu search algorithm for the resource constrained project scheduling problem". *Ribeiro CC, Hansen P, editors: Essays and surveys in metaheuristics, Kluwer Academic Publishers*. pp. 557–88, 2002.

[164] R. Slowinski. "Two Approaches to Problems of Resource Allocation Among Project Activities- A Comparative Study". *Journal of Operational Research Society*, vol. 31, pp. 711–723, 1980.

[165] A. Sprecher and A. Drexl. "Multi-mode resource-constrained project scheduling by a simple, general and powerful sequencing algorithm". *European Journal of Operational Research*, vol. 107, pp. 431–450, 1998.

[166] A. Sprecher, S. Hartmann and A. Drexl. "An exact algorithm for project scheduling with multiple modes". *OR Spektrum*, vol. 19, pp. 195–203, 1997.

[167] L.Y. Tseng and S.C. Chen. "A hybrid metaheuristic for the resource-constrained project scheduling problem". *European Journal of Operational Research*, vol. 175, pp. 707–721, 2006.

[168] V. Valls, F. Ballestn and S. Quintanilla. "A hybrid genetic algorithm for the resourceconstrained project scheduling problem". *European Journal of Operational Research*, vol. 185, pp. 495–508, 2008.

[169] G. Zhu, F.J. Bard and G. Tu. "A Branch-and-Cut Procedure for the Multimode Resource-Constrained Project-Scheduling Problem". *Journal on Computing*, vol. 18(3), pp. 377–390, 2006.

[170] K. Ziarati, R. Akbari and V. Zeighami. "On the performance of bee algorithms for resource-constrained project scheduling problem". *Applied Soft Computing*, vol. 11, pp. 3720–3733, 2011.