THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

Institut National des Sciences Appliquées de Toulouse (INSA de Toulouse)

Présentée et soutenue par :
Panwadee Tangpattanakul

le jeudi 26 septembre 2013

Titre :

Optimisation multi-objectif de missions de satellites d'observation de la Terre

---

École doctorale et discipline ou spécialité :
EDSYS : Informatique 4200018

Unité de recherche :
LAAS-CNRS

Directeur(s) de Thèse :

M. Pierre Lopez, Directeur de recherche CNRS, LAAS Toulouse
M. Nicolas Jozefowiez, Maître de Conférences, INSA Toulouse

Jury :
M. Enrique Alba, Professeur, Université de Málaga, ESPAGNE
M. Frédéric Saubion, Professeur, Université d'Angers
Mme Laetitia Jourdan, Professeur, Université Lille 1
M. Gérard Verfaillie, Directeur de Recherches, ONERA Toulouse

Acknowledgements

This thesis would not have been possible without help and support from all those people. I would like to express my gratitude to all of them.

First, I would like to gratefully and sincerely thank Dr. Pierre Lopez and Dr. Nicolas Jozefowiez, my excellent supervisors, for their support and guidance during these three years of Ph.D. student life. They gave me the freedom to explore on my own. In the same time, when I did not know how I should continue my work, they always guided me to the good ways. Moreover, their patience and understanding helped me to pass the difficult times of my work.

I would like to thank Prof. Enrique Alba and Prof. Frédéric Saubion for the review of my thesis, Prof. Laetitia Jourdan and Dr. Gérard Verfaillie, for accepting to be members of the jury. All of them gave me the interesting suggestions, which I can use to improve my work in the future.

I would like to acknowledge the funding sources, GISTDA (Geo-Informatics and Space Technology Development Agency) of Thailand and the French government for the obtained scholarship that supported all expenses during my Ph.D. study.

Many thanks to all members of ROC group in LAAS-CNRS for their encouragement and the happiness that I always got from them. They made my life in France was more beautiful and I will remember the very good times that we spent together.

I would like to acknowledge all teachers who taught me since my childhood. I would not have been here without their guidance.

Additionally, many thanks to Romaric Guillerm and his family, Dariga and Gaëtan Toulon, Kata Kiatmanaroj, Boadu Mensah Sarpong, Rui Xue, and all of my friends for their valuable help and support.

Finally, I would like to thank my parents who support me for everything what I chose. Thanks to my family who always believes in me, I always get encouragement from them to do the things that I love, until I can finish this thesis.

Panwadee TANGPATTANAKUL
3 October 2013

Abstract

This thesis considers the selection and scheduling problem of observations for agile Earth observing satellites. The mission of Earth observing satellites is to obtain photographs of the Earth surface to satisfy user requirements. Requests from several users have to be managed before transmitting an order, which is a sequence of selected acquisitions, to the satellite. The obtained sequence must optimize two objectives under operation constraints. The first objective is to maximize the total profit of the selected acquisitions. The second one is to ensure the fairness of resource sharing by minimizing the maximum profit difference between users. Two metaheuristic algorithms, consisting of a biased random key genetic algorithm (BRKGA) and an indicator-based multi-objective local search (IBMOLS), are proposed to solve the problem. For BRKGA, three selection methods, borrowed from NSGA-II, SMS-EMOA, and IBEA, are proposed to select a set of preferred chromosomes to be the elite set. Three decoding strategies, which are two single decoding and a hybrid decoding, are applied to decode chromosomes to become solutions. For IBMOLS, several methods for generating the initial population are tested and the neighborhood structure according to the problem is also proposed. Experiments are conducted on realistic instances based on ROADEF 2003 challenge instances. Hypervolumes of the approximate Pareto fronts are computed and the results from the two algorithms are compared.

Résumé

Cette thèse considère le problème de sélection et d'ordonnancement des prises de vue d'un satellite agile d'observation de la Terre. La mission d'un satellite d'observation est d'obtenir des photographies de la surface de la Terre afin de satisfaire des requêtes d'utilisateurs. Les demandes, émanant de différents utilisateurs, doivent faire l'objet d'un traitement avant transmission d'un ordre vers le satellite, correspondant à une séquence d'acquisitions sélectionnées. Cette séquence doit optimiser deux objectifs sous contraintes d'exploitation. Le premier objectif est de maximiser le profit global des acquisitions sélectionnées. Le second est d'assurer l'équité du partage des ressources en minimisant la différence maximale de profit entre les utilisateurs. Deux métaheuristiques, composées d'un algorithme génétique à clé aléatoire biaisées (*biased random key genetic algorithm* - BRKGA) et d'une recherche locale multi-objectif basée sur des indicateurs (*indicator-based multi-objective local search* - IBMOLS), sont proposées pour résoudre le problème. Pour BRKGA, trois méthodes de sélection, empruntées à NSGA-II, SMS-EMOA, et IBEA, sont proposées pour choisir un ensemble de chromosomes préférés comme ensemble élite. Trois stratégies de décodage, parmi lesquelles deux sont des décodages uniques et la dernière un décodage hybride, sont appliquées pour décoder les chromosomes afin d'obtenir des solutions. Pour IBMOLS, plusieurs méthodes pour générer la population initiale sont testées et une structure de voisinage est également proposée. Des expériences sont menées sur des cas réalistes, issus d'instances modifiées du challenge ROADEF 2003. On obtient ainsi les fronts de Pareto approximés de BRKGA et IBMOLS dont on calcule les hypervolumes. Les résultats de ces deux algorithmes sont comparés.

Contents

List of publications

- P. Tangpattanakul, N. Jozefowiez, and P. Lopez. Multi-objective Optimization for Selecting and Scheduling Observations by Agile Earth Observing Satellites. In Coello Coello C.A. et al., editors, *Parallel Problem Solving from Nature - PPSN XII*, volume 7492 of *Lecture Notes in Computer Science (LNCS)*, pages 112-121. Springer Berlin Heidelberg, 2012.

- P. Tangpattanakul, N. Jozefowiez, and P. Lopez. Biased Random Key Genetic Algorithm with Hybrid Decoding for Multi-objective Optimization. In *Proceedings of Federated Conference on Computer Sciences and Information Systems 2013*, Krakow, Poland, September 2013.

- P. Tangpattanakul, N. Jozefowiez, and P. Lopez. Optimisation multi-objectif des prises de vues d'un satellite agile d'observation de la Terre. In *Proceedings of ROADEF 2012*, Angers, France, April 2012.

- P. Tangpattanakul, N. Jozefowiez, and P. Lopez. Multi-objective optimization for Earth observation scheduling. In *Proceedings of ROADEF 2013*, Troyes, France, February 2013.

Introduction

This thesis is in the multi-objective optimization area. It is applied to solve scheduling problems in space applications, especially for Earth observing satellites. The research has been done in the Modeling, Optimization and Integrated Management of Systems of Activities (MOGISA) team of the Laboratory for Analysis and Architecture of Systems (LAAS-CNRS). This study is supported by the THEOS Operational Training Programme (TOTP) of Geo-Informatics and Space Technology Development Agency - Thailand (GISTDA). For the publications, some parts of this work had been presented in the Annual Congress of the French Society of Operations Research and Decision Support (ROADEF 2012 and ROADEF 2013), and the International Conference on Parallel Problem Solving from Nature (PPSN 2012). Furthermore, the latest paper was accepted to be presented in the Workshop on Computational Optimization (WCO 2013).

The mission of Earth observing satellites is to obtain photographs of the Earth surface depending on requests from users, in order to satisfy their requirements. The requests must be managed by the ground station center before transmitting a sequence to operate the satellites. In this work, we study the multi-user observation scheduling problem of an agile Earth observing satellite. Requests required from several users are considered. For a spot request, it can be acquired at once by the satellite, hence it is considered as a strip. For a polygonal request, it has to be decomposed into several strips. Each strip can be acquired in two opposite directions, called possible acquisitions. However, only one direction can be selected. The problem needs to find a sequence of selected acquisitions, which optimizes two objective functions and satisfies the satellite constraints. The first objective function is to maximize the total profit. The second one is to ensure the fairness of resource sharing by minimizing the maximum profit difference between users. Therefore, the problem is modeled as a multi-objective optimization problem.

For years, many real-world optimization problems were looked and

treated as multi-objective optimization problem, thus a numerous number of multi-objective optimization algorithms were proposed. The family of metaheuristics is one of many existing methods, which were applied to solve problems in engineering and scientific areas. Metaheuristics consist of a panel of approximate optimization methods, which are able to find near optimal solution in a reasonable time. They are usually efficient to obtain good solutions for solving complex problems.

In this work, two metaheuristic algorithms, which are a biased random key genetic algorithm (BRKGA) and an indicator-based multi-objective local search (IBMOLS), are applied to solve the multi-user observation scheduling problem of an agile observing satellite. Some steps of the algorithms are adapted from the original proposed version for solving the considered problem.

The document of this thesis is organized as follows: Chapter 1 presents an overview of multi-objective optimization. Its basis concepts and mathematical formulation are explained. Since multi-objective optimization considers several objectives simultaneously, we will not obtain only one solution, but a set of solutions. Thus, dominance relation, tradeoff surface, and approaches for fitness assignment are covered. Some metaheuristics, which are developed to solve multi-objective optimization problems, are also described in this chapter.

Chapter 2 describes the observation scheduling problem for Earth observing satellites. Two types of Earth observing satellites, agile and non-agile, are studied in this work. The models of their observation scheduling problems are explained. The scheduling problem concerning an agile satellite is more complex to solve, because of its properties. Hence, the multi-objective scheduling problem of an agile satellite is mainly considered in this work. The modified instances of ROADEF 2003 challenge, which will be used for testing the algorithms, are also described in detail.

In Chapter 3, BRKGA is applied to solve the problem. Since two objectives are considered in this work, the selection methods from three multi-objective evolutionary algorithms are used to choose a set of preferred chromosomes to become the elite set in BRKGA process. Moreover, a hybrid decoding is proposed to improve the results obtained from single decodings.

IBMOLS is applied to solve the problem in Chapter 4. Three methods of initial population generation are presented. Moreover, neighborhood structures and strategies for checking the feasibility of solutions are also described.

Finally, experimental results of the two algorithms are reported and discussed in Chapter 5. Hypervolumes of the obtained approximate Pareto fronts are computed. The hypervolume values and computation times are compared. Moreover, an example of the approximate Pareto front of each instance is also illustrated in this chapter. The document finishes by highlighting the conclu-

sions of the document, as well as proposing some short-term perspectives and possible further research directions.

1 Multi-objective optimization

This chapter presents an overview of multi-objective optimization. Many real world optimization problems consider several objectives simultaneously. Hence, they can be solved by using multi-objective optimization algorithms. Not only one solution, but a set of solutions, will be obtained after solving the problem. A subset of these solutions is interesting and they can be decided by the dominance relation. In this work, some metaheuristics are applied to solve the problem and their concepts are described in this chapter.

1 Introduction

Optimization is an important issue for many real-world applications. Many areas are concerned, e.g. engineering, science, management, business, etc. The optimization involves the decision making which chooses a best solution from all alternative available choices. The fitness measurement of each choice depends on a given objective function. A general optimization problem can be formalized as:

$$\begin{aligned} &\text{minimize} && f(x) \\ &\text{subject to} && x \in \Omega \end{aligned}$$

where $f : \mathbb{R}^n \rightarrow \mathbb{R}$, $x = (x_1, x_2, \ldots, x_n) \in \Omega$, and $\Omega \subseteq \mathbb{R}^n$. The function f is an objective function that we need to minimize. The vector x is an n-vector of decision variables. The set Ω is a feasible set. Furthermore, note that a maximization objective can be represented as a minimization one ($minimize f(x) \equiv maximize - f(x)$) [17]. For obtaining the best solution of the optimization problem, many algorithms, which use different searching strategies, were proposed. All strategies of mono-objective optimization algorithms have a common goal, which needs to find the global optimum of

the objective function while satisfying the constraints. Some examples of classical algorithms are the sequential search [54], Newton-Raphson method [87], gradient method [32], or simplex method [66]. Examples of evolutionary algorithms, which use mechanisms inspired by biological evolution, are genetic algorithm [43], ant colony optimization [28], particle swarm optimization [53], or harmony search algorithm [39]. Some other metaheuristic algorithms are simulated annealing [55], or tabu search [42].

Real world problems are not usually characterized by only one objective, as there are several conflicting objectives to handle. Hence, a multi-objective optimization is more efficient to address the global problems.

2 Basis concepts and mathematical formulation

For the history of multi-objective optimization, Olivier L. De Weck said in [24] *"Francis Y. Edgeworth and Vilfredo Pareto are credited for first introducing the concept of non-inferiority in the context of economics "*. After that, many engineering researchers were interested in the area of multi-objective optimization and it was rapidly developed until nowadays. For instance, J. Andersson reformulated the design problem as a multi-objective optimization problem in [3]. It is obviously more natural to express a real world question as a multi-objective optimization problem. However, the methods for solving multi-objective optimization problems are much more complicated. Therefore, efficient optimization strategies are required [89]. A simple instance of a multi-objective optimization problem is the following: Consider people who want to buy a product. They need the product with the highest quality. Thus the product quality has to be maximized. Simultaneously, they do not want to pay a lot of money, hence the product price has to be minimized. The example set of favorable solutions is shown in Figure 1.1. For the bi-objective optimization problem, normally both objective functions are contradictory. As in the example, when the people increase the quality of the product, its price will not be cheap.

A multi-objective optimization problem has a general form as follows:

$$\begin{aligned} &\text{minimize} && F(x) = (f_1(x), f_2(x), \ldots, f_n(x)) \\ &\text{subject to} && x \in \Omega \end{aligned}$$

where:

- $n \geq 2$ is the number of objectives,
- $F = (f_1, f_2, \ldots, f_n)$ is a vector of functions to optimize,
- $\Omega \subseteq \mathbb{R}^m$ is a set of feasible solutions,

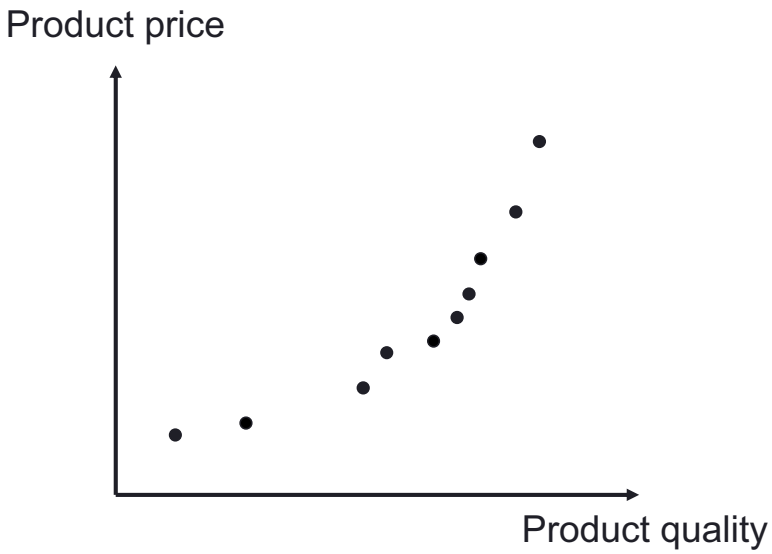


Figure 1.1: The example set of favorable solutions for multi-objective optimization problem.

- $A(\Omega) \subseteq \Omega$ is a set of feasible solutions visited by an algorithm A
- $x = (x_1, x_2, \ldots, x_m) \in \Omega$ is a solution
- $\mathcal{Y} = F(\Omega)$ is the objective space, and
- $y = (y_1, y_2, \ldots, y_n) \in \mathcal{Y}$ with $y_i = f_i(x)$ is a point of the objective space.

Normally, when the optimization problem is solved, we hope to obtain only the best solution. However, for the multi-objective optimization problem, we will not obtain only one solution, but a set of solutions. In a multi-objective optimization problem (considering several contradictory objective functions), the obtained tradeoff solutions are called Pareto-optimal solutions.

3 Dominance relation and tradeoff surface

Among a possible huge number of solutions arising from the solving of a multi-objective optimization problem, only a subset of these solutions is interesting. For characterizing this subset, the following Pareto dominance holds [20]:

Definition 1.1. *A vector x_1 dominates a vector x_2, if x_1 is at least as good as x_2 for all objectives, and x_1 is strictly better than x_2 for at least one objective.*

The solutions which dominate the others and are not dominated by anyone of them, are called optimal solutions in the Pareto sense or nondominated solutions. This dominance relation is illustrated in Figure 1.2. The multi-objective optimization problem at hand has to maximize the product quality $f_1(x)$ and minimize the product price $f_2(x)$. Solutions A, B, C, D, and E are found in the solution space. Due to the problem which needs to maximize f_1 and minimize f_2, the dominance relation in the Pareto sense of this problem is given by:

solution x_1 dominates solution x_2 (denoted by $x_1 \prec x_2$) if
$f_1(x_1) \geq f_1(x_2)$ and $f_2(x_1) < f_2(x_2)$
or
$f_1(x_1) > f_1(x_2)$ and $f_2(x_1) \leq f_2(x_2)$.

Hence, the nondominated solutions are solutions A, C, and E, as shown in Figure 1.2.

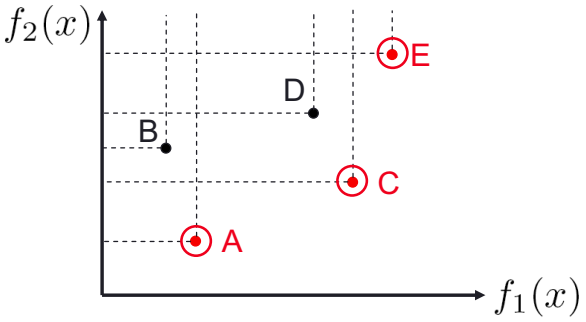


Figure 1.2: Example of nondominated solutions for the multi-objective optimization problem which has to maximize the first objective and minimize the second objective.

There is other types of dominances, which are:

Definition 1.2. *A vector x_1 weakly dominates a vector x_2 (denoted by $x_1 \preccurlyeq x_2$) if x_1 is better than or equal to x_2 for all objectives.*

Definition 1.3. *A vector x_1 strictly dominates a vector x_2 (denoted by $x_1 \prec\prec x_2$) if x_1 is better than x_2 for all objectives.*

The set of points on the objective space, which are mapped from the non-dominated solutions is called tradeoff surface or Pareto front. For the above example, which needs to maximize the first objective and minimize the second objective, the tradeoff surface is illustrated in Figure 1.3.

The shape of the tradeoff surface depends on the type of the problem. For the bi-objective optimization problems, the different shapes of tradeoff surface are shown in Figure 1.4 [20], [67]. For most multi-objective optimization

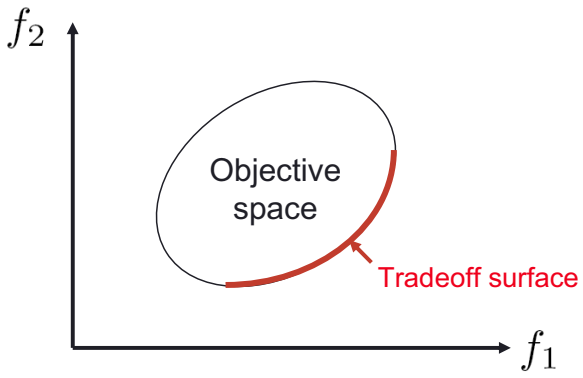


Figure 1.3: Example of tradeoff surface for the multi-objective optimization problem, which has to maximize the first objective and minimize the second objective [20].

problems the exact tradeoff surface cannot be reached for complexity reasons. Thus, the obtained solutions are usually an approximate tradeoff surface. The good approximate tradeoff surface should be as close as possible to the exact tradeoff surface. Furthermore, it should also spread uniformly along the exact tradeoff surface.

4 Approaches for fitness assignment

Several approaches can be used to solve multi-objective optimization problems to optimality. These approaches mainly differ by the way for classifying the quality of the obtained solutions. Four types of approaches are presented in this section: nonscalar approach, scalar approach, Pareto approach, and indicator-based approach.

4.1 Nonscalar approach

The first type of methods refers to nonscalar ones. A classical method, which uses the nonscalar approach, is VEGA (vector evaluated genetic algorithm) proposed in [71]. It applies a selection mechanism differently than in a classical genetic algorithm. In the selection step, a modified version of current population is generated. It is composed of several subpopulations equal to the number of objective functions. The individuals in subpopulation i, are selected from the current population by considering only the objective function f_i. Hence, each subpopulation selects the individuals independently from the other ones. Before starting the crossover and mutation steps, all individuals in

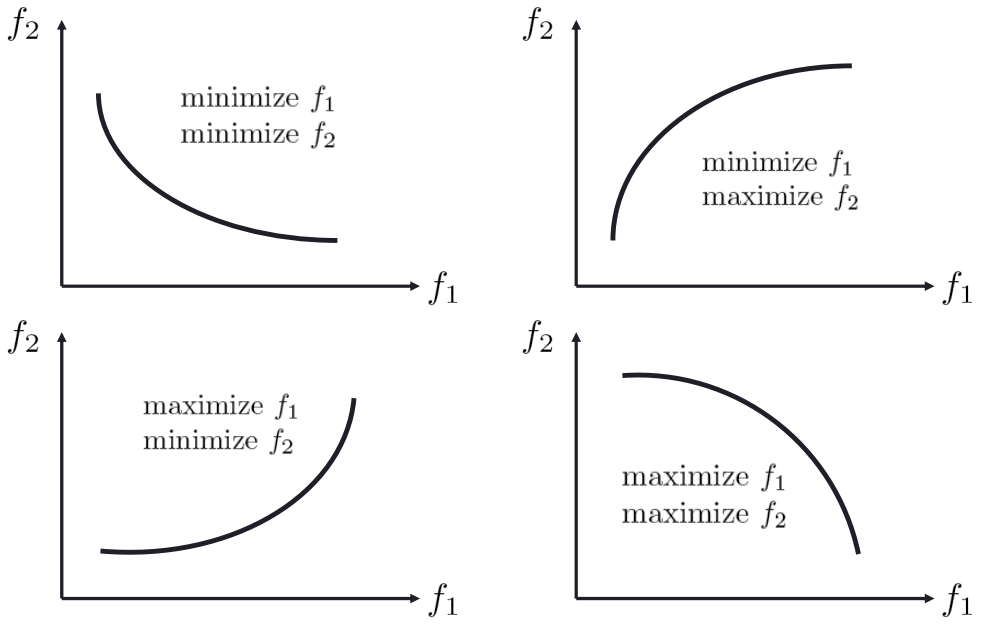


Figure 1.4: The different shapes of tradeoff surface for bi-objective optimization problem [20].

each subpopulations are mixed in order to obtain the modified current population. Then, crossover and mutation mechanisms can be applied. The VEGA process is shown in Figure 1.5. However, this algorithm has a main disadvantage, that when the algorithm selects the individuals to keep in each subpopulation, it considers only one objective without looking the others. Thus, the compromised solutions, which consider all objectives, can be lost.

4.2 Scalar approach

Some cases of multi-objective optimization problems can be solved by using the scalar approach. The main target of the scalar approach is to transform the multi-objective problem into the mono-objective problem. Many algorithms were proposed by using this scalar approach, e.g., the aggregation or weighted sum of objective functions, the weighted matrix, the ϵ-constraint method, etc.

For example, when using the weighted sum of objective functions method, which transforms several objectives into a single one, the optimization prob-

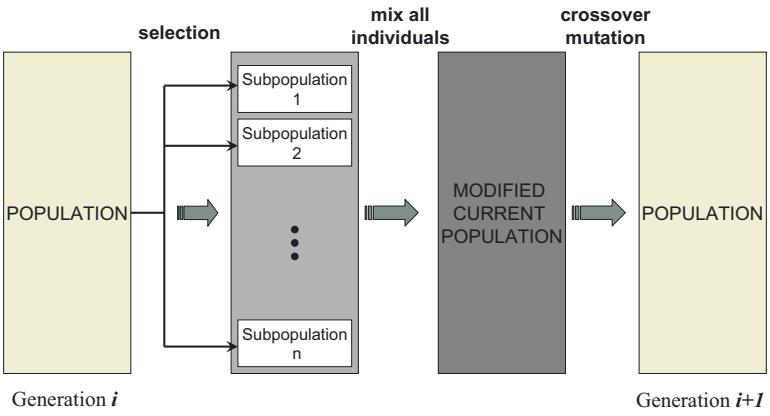


Figure 1.5: VEGA process.

lem is given by

$$\begin{aligned} &\text{minimize} \quad f_{eq}(x) = \sum_{i=1}^{n} w_i \cdot f_i(x) \\ &\text{subject to} \quad x \in \Omega \end{aligned}$$

where n is the number of objective functions, w_i is the weight of objective function f_i, and Ω is a solution space. Then, the mono-objective optimization method can be used for solving this problem. However this method has a disadvantage which is how to define the weight values, because they have to be assigned by users.

4.3 Pareto approach

The idea of Pareto approach or dominance-based approach was presented firstly in [43]. It uses the dominance relation as presented in Section 3 for the fitness assignment of each solution. This approach does not need to transform the multi-objective optimization problem into the mono-objective optimization problem. Moreover, a set of solutions, which converge to the set of Pareto optimal solutions, are obtained in a single run. The obtained solutions are the nondominated solutions, which are classified by using the dominance relation. Because of this advantage, the Pareto approach is used in many multi-objective optimization algorithms. Examples of famous algorithms using the Pareto approach, are strength Pareto evolutionary algorithm 2 (SPEA2) and nondominated sorting genetic algorithm II (NSGA-II). Both algorithms will be detailed in Sections 6.2 and 6.3, respectively.

4.4 Indicator-based approach

The indicator-based approach uses the binary indicator I for giving the quality of the found solutions. There are several types of indicators, for example, ϵ-indicator or hypervolume indicator. The binary indicator $I(A, B)$ can be calculated for comparing the quality between two sets of solutions A and B. The indicator-based approach has some advantages. For example, it does not require the diversity preservation mechanism such as fitness sharing. Also, it does not need to define the values of weighted parameters as in some scalar approaches. Because of these advantages, the indicator-based approach can be easily included in the optimization algorithms in order to develop the preference selection step in the algorithms. The indicator-based approach is used in many multi-objective optimization algorithms, for instance, indicator-based evolutionary algorithm (IBEA), $\mathscr{S}$ *metric selection* evolutionary multi-objective optimization algorithm (SMS-EMOA), or indicator-based multi-objective local search (IBMOLS). These three algorithms will be described in detail in Sections 6.4, 6.5, and 6.6, respectively.

The presented approaches for fitness assignment measure the quality of the solutions found by multi-objective optimization algorithms. They are applied to select and preserve the good solutions and discard the bad ones.

In the next section, the metaheuristic algorithms will be presented. They consist of many efficient methods to solve real-world problems.

5 Metaheuristics

Solving real-world problems, involving conflicting objective functions, is characterized by the huge size of search space. Eckart Zitzler said in [89] *"the search space of the multi-objective optimization problem can be too large and too complex to be solved by exact methods "*. Thus, another type of solving methods, so-called metaheuristics, will be presented in this section. The metaheuristics represent a family of approximate optimization techniques [77]. They are able to find approximate optimal solutions in a reasonable time. Most of them iteratively improve the quality of the candidate solutions. The metaheuristics obtain approximate solutions, which cannot be guaranteed to be optimal as exact methods do. However, metaheuristics are usually efficient to obtain good solutions for solving complex problems in engineering and scientific areas.

Heuristics are techniques for seeking good solutions at a reasonable cost [83]. They are different from the metaheuristics in the sense that the heuristics are ad hoc, i.e., designed for specified problem, while the metaheuristics are general methods, which can be applied to solve several different problems. Thus, the metaheuristics are defined as upper level methodologies [77]. As

in [49], the metaheuristics were proved to be highly efficient to solve various difficult combinatorial optimization problems. In [15], the importance of metaheuristics to solve combinatorial optimization problems was discussed. Moreover, an overview and a comparison of the main metaheuristic methods are presented. Furthermore, the metaheuristics, especially population-based ones, are the main techniques to solve multi-objective problems. Surveys, tutorials and state-of-the-art research papers in this field were presented in [37]. Several metaheuristic methods were proposed since the early 1950s and some of them are described below.

5.1 Local search

Local search is a metaheuristic method which improves a single solution. It starts by an initial solution in the search space. Then, a neighborhood structure is defined depending on the problem, in order to find a new solution, which can improve the objective function value from the incumbent one. In each iteration, the neighbors are explored and the current solution is replaced by an improving neighbor. The iterations will be continued until no better neighbors can be found. It means that a local optimum is reached. In [23], the local search was first applied to solve combinatorial optimization problems. A local search approach was implemented in [51] to the particular case of the traveling salesman problem.

Some important questions when using local search are how to pick an initial solution, how to define the neighborhood, how to select the neighbor for replacing the current solution. Moreover, the main drawback of local search is that it may converge rapidly to a local optimum while the global optimum cannot be reached. For the first question, there are several methods for picking the initial solution, e.g. starting from a greedy initial solution in [31] or starting from just one random feasible solution in [70]. For the second question, the structure of the neighborhood can be defined depending on the applied problems. However, the size of neighborhood has an effect for finding the final solution. On the one side, if the neighborhood is defined in the too large area, the final solution can be the global optimum, but it may require a very high computation time. On the other side, a too small neighborhood is defined, and the final solution is probably only a local optimum. There are many researches which studied the algorithms to define the neighborhoods, e.g. a survey of very large-scale neighborhood search techniques [1], a proposed neighborhood portfolio approach which were applied to timetabling problems [38]. For the third question about how to select the neighbor for replacing the current solution, the selection of the neighbor can apply from many strategies [77]:

- The best improvement works by exploring all neighbors in the neighborhood and choosing the best one of them.
- The first improvement selects the first explored neighbor which is better than the current solution.
- The random selection chooses randomly a neighbor from the set of improving ones, after all neighbors are explored.

Due to the three questions when using the local search, several parameters have to be set properly for operating an efficient algorithm. An efficient algorithm should avoid to entrap local optima and reach the global optimal solution. Some algorithms were proposed to improve the basic local search. For instance, an iterated local search is presented and generalized in [63]. This method repeats the basic local searches from the different initial solutions. In each iteration, the initial solution is generated by using the perturbation. It is more efficient than using the initial solutions which are regenerated randomly. Moreover, this method also uses an acceptance criterion to decide whether a neighboring solution is accepted or not. This criterion improves the diversification.

5.2 Genetic algorithm

Genetic algorithm is a population-based method which was first applied for solving the optimization problems in [43]. It is one of the very popular evolutionary algorithms which is developed from the natural survival concept. The genetic algorithm operates by several individuals in the population. Each individual consists of a chromosome, which represents one solution of the search space. The types of chromosome encoding, e.g. binary, real value, or permutation, are defined depending on the considered problem. A genetic algorithm is an iterated method which is started by generating an initial population and its size equals to p. Three important operations, which consist of selection, crossover, and mutation, are used to generate the new population for the next generations. The crossover and mutation operators are operated according to their probabilities. The generation process is repeated until some stopping criteria are satisfied.

For the selection operator, the individuals that exist in the population are selected by considering the fitness of each individual, which is obtained after the chromosome decoding. The fittest individuals are chosen to survive and they are stored into the population in the next generation. The crossover operator is used to produce the offspring individuals (children) for the next generation. Two parents are selected from the current population, then an offspring is generated from them. There are many techniques to crossover, e.g.

one-point crossover, two-point crossover, or uniform crossover. The last operator is mutation operator, which is used for escape from a local optimum. The mutation operator has several techniques, e.g. bit flip at a random position or regenerating as the initial individuals. However, the probability of mutation should be set low, otherwise, the search will become a random search.

Genetic algorithms were applied to solve several well-known problems in the optimization area [72]. Genetic algorithms were also used for solving multi-objective optimization problems for a long time [78]. Several efficient methods were proposed by using different strategies [56]. Multi-objective optimization algorithms based on genetic algorithm will be discussed in Section 6.

In this section, we will introduce a special type of genetic algorithm, which is applied for solving the problem under study. It is a biased random key genetic algorithm (BRKGA).

Biased random-key genetic algorithm

The biased random-key genetic algorithm (BRKGA) was first presented in [44]. It is one efficient method for solving combinatorial optimization problems. The BRKGA was developed from random key genetic algorithm (RKGA), which was proposed in [9]. The BRKGA has a different way to select two parents for the crossover operation, when comparing with the seminal RKGA. The BRKGA selects randomly a parent from an elite set and another parent from a non-elite set of the current population. On the contrary, the original RKGA randomly selects two parents from the entire population. The BRKGA combines genetic algorithm with the concept of random key. The random-key chromosome, which is formed by several genes, represents one solution. As the basic process of genetic algorithm, the BRKGA has the same operating steps. It is started by generating an initial population and the population consists of p chromosomes. The genetic algorithm involves three mechanisms: selection, crossover, and mutation, to generate the new chromosomes for the next generation. The iterations are repeated until the stopping criteria are satisfied.

For BRKGA, the random key chromosome is formed by several genes that are encoded by real values randomly generated in the interval $[0, 1]$. Then, it inputs the chromosome to a decoder in order to output the solution. The decoder is defined depending on the problem. For the scheduling problem, each gene represents one job. The solution stands for a sequence of jobs. The priority to consider each associated job in order to be assigned in the sequence is calculated in this decoding step. Objective function values and fitness of the solution, which is obtained from each chromosome, can be computed. To start the process for generating the new generation, the current population is classified into two groups by using the selection mechanism. The methods

for selection are applied to choose p_e preferred chromosomes from the current population to become the elite set. The remaining chromosomes will be stored in the other group of non-elite chromosomes. Then, the process to generate the population in the next iteration is begun.

The population of the new generation is generated from three parts, as in Figure 1.6. The p_e chromosomes of elite set are copied to be stored in the top position. The second part is stored in the bottom position, which is a mutant set. It is the set of p_m chromosomes generated to avoid entrapment in a local optimum. These chromosomes are randomly generated by the same methods used to generate the initial population. The last part is the crossover part for which each crossover offspring is built from one elite chromosome and one chromosome in the previous population. Each element in the crossover offspring is obtained from the element in elite chromosome with the probability ρ_e. The crossover offspring is stored in the middle part of the new population. Hence, the size of crossover offspring set is $p-p_e-p_m$ to fulfill the remaining space of chromosomes in the next population. The recommended parameter value setting is displayed in Table 1.1 [45]. The process for generating the next populations is applied repeatedly until a given stopping criterion is satisfied.

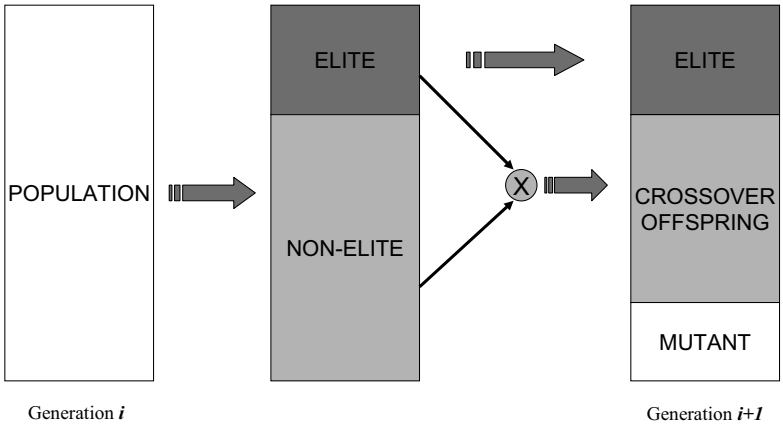


Figure 1.6: The population of the new generation by using BRKGA.

The BRKGA was used to solve several combinatorial optimization problems (e.g. communication, transportation or scheduling) [45]. In [46], the authors used BRKGA to solve the fiber installation for an optical network optimization problem. The objective function is to minimize the cost of the optical components necessary to operate the network. In [65], the resource-constrained project scheduling problem was solved by BRKGA. They found the optimal solution for minimizing the makespan. Nevertheless, all of these problems need to find an optimal solution for satisfying a single objective

Table 1.1: Recommended parameter values of BRKGA [45].

Parameter	Recommended value
p	$p = a.n$, where $1 \leq a \in \mathbb{R}$ is a constant and n is the length of the chromosome
p_e	$0.10p \leq p_e \leq 0.25p$
p_m	$0.10p \leq p_m \leq 0.30p$
ρ_e	$0.5 \leq \rho_e \leq 0.8$

function. In our work, the BRKGA is applied to solve multi-objective optimization problems. This will be presented in Chapter 3.

5.3 Other metaheuristic algorithms

In Sections 5.1 and 5.2, the metaheuristics, which are applied to solve the problem in this work, are presented. Except the local search and the genetic algorithm, there are other interesting metaheuristic algorithms and some of them will be discussed in this section.

Simulated annealing was first proposed to solve optimization problems in [55]. It corresponds to a probabilistic method for finding the global minimum of an objective function that may have several local optima [11]. This method bases on the annealing process, where a solid material is heated and then it is cooled down slowly in order to obtain a strong structure. The simulated annealing algorithm applies the energy change of the cooling down process until it converges to an equilibrium state for solving optimization problems. The practical guideline for the implementation of simulated annealing was presented in [48]. In [76], [5], the simulated annealing is developed to provide a set of tradeoff solutions for solving multi-objective optimization problems.

Ant colony optimization belongs to swarm intelligence methods and is also a probabilistic technique for solving optimization problems. It is one of the very famous metaheuristics in the combinatorial optimization area. It was successful for solving several well-known problems, e.g. traveling salesman problem [27], permutation flowshop scheduling [68], and so on. The ant colony optimization algorithm was first proposed in [26]. This algorithm was established from an inspiration of ant's behavior, when the ant seeks the shortest path for transporting the food between its nest and the food source. The ant colony algorithm consists of two main iterated steps, which are a solution construction by using the pheromone trail and a pheromone update of evaporation and reinforcement. The ant colony optimization algorithm is applied for solving problems in several applications [29], [30]. It was initially

proposed for solving single objective combinatorial optimization problems. After that, it was adapted to be able to solve continuous optimization problems [73] and multi-objective optimization problems [2].

Particle swarm optimization refers also to swarm intelligence. It is a population-based metaheuristic. It was first presented in [53]. This method mimics the social behavior of natural organism which is the movement of organisms in a bird flock or fish school to find a food place. It optimizes the problems by using the population of the particles as the candidate solutions. Particle swarm optimization was applied to solve optimization problems in many areas [74], [6]. It was also developed for multi-objective optimization problems [50], [18].

In the next section, metaheuristics, especially evolutionary algorithms, used to solve multi-objective optimization problems, will be presented.

6 Metaheuristics for multi-objective optimization

Many metaheuristics were developed for solving multi-objective optimization problems. This work considers the algorithms in the evolutionary class, as well as local search algorithms. The evolutionary class is a class of metaheuristics which simulates the process of natural evolution. The techniques of this class are popular due to the effectiveness and robustness in searching for global tradeoff solutions [79]. This section will discuss about evolutionary and local search algorithms for multi-objective optimization.

Firstly, the multi-objective evolutionary algorithms (MOEA) will be introduced in Sections 6.1 – 6.5. As presented previously about the good approximate Pareto front of the multi-objective optimization, the goals are to minimize the distance between the obtained solutions of the approximate Pareto front and the exact one and to maximize the diversity of the approximate Pareto front. Thus, algorithm design issues are fitness assignment, diversity preservation, and elitism [93]. For the first issue, the main role of fitness assignment is to guide the search toward to the exact solutions. The fitness is a scalar value which can illustrate the quality of the objective function vector. For the second one, the goal of diversity preservation is to keep the diverse set of nondominated solutions. Finally, elitism helps the algorithm to keep the nondominated solutions during the optimization process. Most researches on MOEA concentrate on the selection stage which needs to integrate vectorial performance measures in order to classify the solution qualities [34]. Several researchers developed many well-known algorithms and some of them will be described in this section.

6.1 Multiple objective genetic algorithm (MOGA)

Multiple objective genetic algorithm or MOGA was proposed in [33]. This method is based on the dominance relation in the Pareto sense. The fitness is assigned to each individual depending on the number of solutions of which it is dominated by. For instance, consider an individual x_i at generation t which is dominated by $p_i^{(t)}$ individuals, the rank of this considered individual is given by rank $(x_i, t) = 1+p_i^{(t)}$. All nondominated solutions are assigned rank 1. The other ones are assigned higher ranks according to the population density of the corresponding region. For this algorithm, not all ranks will necessarily be represented in the population. Their ranks are transformed to the fitnesses by interpolating from the best rank (rank 1) to the worst rank (rank n^*) according to a function which is often linear as illustrated in Figure 1.7. The fitness assignment in this method has an important drawback. It lacks to consider the diversity of the individuals. Therefore, niche-formation method is used to distribute the population by performing a sharing parameter. However, the performance of this algorithm is highly dependent on an appropriate selection of the sharing factor [19].

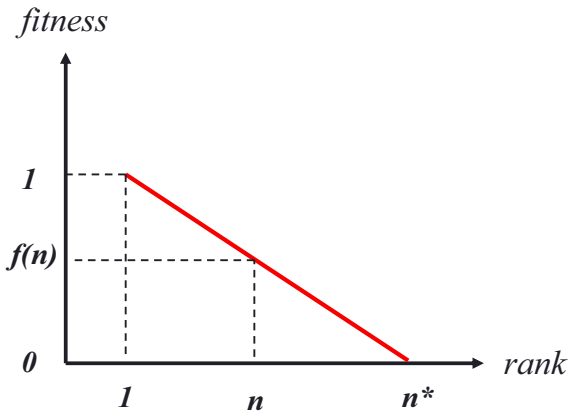


Figure 1.7: Fitness assignment for MOGA by using linear interpolation.

6.2 Strength Pareto evolutionary algorithm 2 (SPEA2)

Strength Pareto evolutionary algorithm 2 (SPEA2) was proposed in [90]. It is the improved version of SPEA [91]. SPEA uses a population P and an archive A. The archive is used to store the nondominated solutions and it is updated when finishing the process in each iteration. Furthermore, the size of archive is limited by a predefined value. In the algorithm process, the fitness is assigned to each solution, both in archive and population. The fitness of

each solution in the archive $(y \in A)$ is calculated depending on its strength. The strength values are in the interval $[0, 1[$. The strength value of solution y for iteration t is defined as

$$s(y,t) = \frac{np(y,t)}{N_p + 1},$$

where $np(y,t)$ is the number of solutions in the population, which do not dominate solution y and N_p is the population size. For each solution in the regular population $(x \in P)$, the fitness is given by

$$f(x,t) = 1 + \sum_{y \in A, y \preccurlyeq x} s(y,t)$$

That way, all nondominated solutions are assigned a fitness equal to 1. However, the nondominated solution which has the lowest strength value is preferred than the others, since it dominates the least number of solutions in the objective function space. This value can indicate the diversity of the solutions. Hence, the solution with the lower fitness value has more chance to be selected. Afterwards, the crossover and mutation mechanisms are applied.

However, SPEA has some issues that need to be improved. The first issue is about the fitness assignment; solutions, which are dominated by the same archive members, have the same fitness values. Hence, the selection pressure is decreased substantially. For the second one, in case many solutions are indifferent, density information has to be used in order to guide the search more effectively. The last one is about the archive truncation. Because of the limitation of archive size, SPEA may lose some good nondominated solutions [90]. Hence, SPEA2 was proposed for improving the algorithm. The main procedure of SPEA2 has the steps as in Algorithm 1.

Both SPEA and SPEA2 are very efficient to solve multi-objective optimization problems. They are also good examples for using the archive set to store the nondominated solutions during the search [56]. Especially for SPEA2, it is applied to solve the problems in many applications until nowadays.

6.3 Nondominated sorting genetic algorithm II (NSGA-II)

Nondominated sorting genetic algorithm II (NSGA-II) was proposed in [25]. It is an improved version of NSGA which was proposed in [75]. For NSGA, the individuals in the population are classified into several layers of classifications. NSGA uses the dominance relation in the Pareto sense for ranking the individuals. All nondominated individuals are classified in one category with the dummy fitness value [19]. The process continues until all individuals in the population are classified. NSGA has a computational complexity of

Algorithm 1 Procedure of SPEA2

N_A is the archive size
N_P is the population size
U_A is the maximum size of archive set A

Step 1: Initialization
Set $t = 0$.
Generate an initial population P_0 and an empty archive set $A_0 = \emptyset$.

Step 2: Fitness calculation

for all $x \in P_t \cup A_t$ **do**

Step 2.1: Calculate the raw fitness $r(x,t) = \sum_{y \in P_t \cup A_t, y \prec x} s(y,t)$, where $s(y,t)$ is the number of solutions in $P_t \cup A_t$ dominated by solution y.
Step 2.2: Calculate the density $d(x,t) = 1/(\sigma_x^k + 2)$, where σ_x^k is the distance between solution x and its k-th nearest solution point, and $k = \sqrt{N_A + N_P}$.
Step 2.3: Calculate the final fitness $f(x,t) = r(x,t) + d(x,t)$.
end for

Step 3: Archive set management
$A_{t+1} \leftarrow$ nondominated solutions in $P_t \cup A_t$
if $|A_{t+1}| < U_A$ **then**
$A_{t+1} \leftarrow$ best $U_A - |A_{t+1}|$ dominated solutions from $P_t \cup A_t$
else
if $|A_{t+1}| > U_A$ **then**
Remove $|A_{t+1}| - U_A$ nondominated solutions from A_{t+1}
end if
end if

Step 4: Termination
if Stopping criteria are satisfied **then**
return Nondominated solutions in A_{t+1}
Stop
end if

Step 5: Selection
Select parents from A_{t+1} by using binary tournament selection with replacement.

Step 6: Crossover and mutation
Apply crossover and mutation operators in order to create the offsprings.
$P_{t+1} \leftarrow$ offsprings
$t \leftarrow t + 1$
Go to Step 2

$O(MN^3)$, where M is the number of objectives and N is the population size. It makes NSGA computationally expensive for large population sizes. For diversification, NSGA uses the concept of sharing, which requires the specification of a sharing parameter. NSGA has a good convergence, however, there are three criticisms which consist of a high computational complexity of nondominated sorting, a lack of elitism which prevents the loss of good evaluated solutions, and a need for specifying the sharing parameter. Hence, NSGA-II was proposed for improving the algorithm. The main procedure of NSGA-II has the steps as in Algorithm 2.

The fast nondominated sorting algorithm is used for ranking the individuals into several fronts. It compares each individual p with other individuals in the population. For each individual p, an integer value n_p and a set S_p are computed. The value n_p illustrates the number of solutions which dominate p. The set S_p stores the individuals which are dominated by p. The process to classify the individuals into each front (starting from $i = 1$) are repeated by checking the n_p values. The individuals, which have n_p equal to 0, are classified by moving them to the considered front $(\mathcal{F}_i)$. Before checking for the next front, n_p and S_p of the remaining individuals are updated. The front classifying are iterated until the individuals in the population are empty. Finally, all individuals in the population are sorted in several ranks, as shown in Figure 1.8.

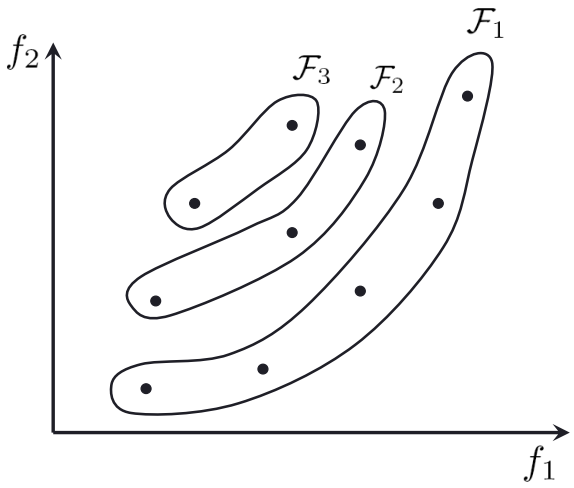


Figure 1.8: Fast nondominated sorting for NSGA-II of the problem to maximize $f_1(x)$ and minimize $f_2(x)$.

The crowding distance assignment is used for ensuring the diversity in the considered front. It sorts the individuals depending on the distances between the two adjacent points in the objective space as in Figure 1.9. The individuals

Algorithm 2 Procedure of NSGA-II

Step 1: Initialization
Set $t = 0$.
Generate randomly an initial population P_0 with size N.

Step 2: Fast nondominated sorting for initial population
Sort the population P_0 in several ranks by using the fast nondominated sorting algorithm.

Step 3: Selection, crossover, and mutation for initial population
Use binary tournament selection, crossover, and mutation for generating a population of offsprings Q_0 with size N.

repeat
Step 4: Iteration
Step 4.1: Combine parents and offsprings population, $R_t = P_t \cup Q_t$.
Step 4.2: Sort all nondominated fronts of R_t, $\mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2, \ldots)$.
Step 4.3: Set $i = 1$,
repeat
Fill the individuals from front i $(\mathcal{F}_i)$ into the new population P_{t+1}
$i \leftarrow i + 1$
until $|P_{t+1}| + |\mathcal{F}_i| \geq N$
Step 4.4: Crowding distance assignment
if $|P_{t+1}| + |\mathcal{F}_i| > N$ **then**

Calculate the crowding distance of $\mathcal{F}_i$.
Sort the individuals in $\mathcal{F}_i$ depending on the crowding distance values from high to low.
Choose first $N - |P_{t+1}|$ individuals to fill into the new population P_{t+1}.
end if
Step 4.5: Use binary tournament selection, crossover, and mutation for generating a population of offsprings Q_{t+1} with size N.
Step 4.6: Increment a generation counter $t = t + 1$.
until Stopping criteria are satisfied.

are ordered in the sorted set according to the distance values from high to low. For the boundary points, they always stay in first positions of the sorted set, so that they are always selected.

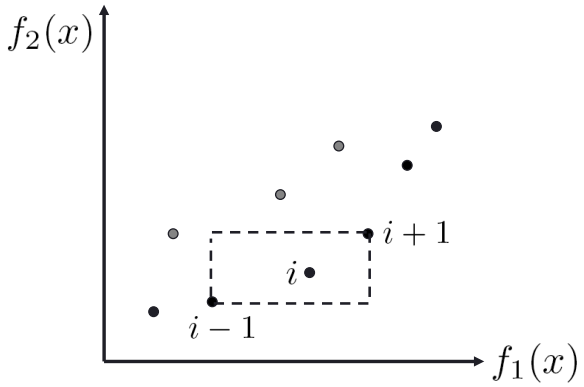


Figure 1.9: Crowding distance assignment for NSGA-II of the problem to maximize $f_1(x)$ and minimize $f_2(x)$.

In the selection step of NSGA-II, two solutions are compared. The solution with a lower nondominated front is preferred. Otherwise, in the case that the two solutions have the same front, the solution which has the higher crowding distance value is preferred. Because this solution has lesser density of solution crowd on the objective space. The overall complexity of NSGA-II is $O(MN^2)$ [25]. The diversity among nondominated solutions uses the crowding distance comparison. It does not need the extra sharing parameter. Because of the efficiency of NSGA-II, it was applied for solving the multi-objective optimization problems in many applications. It is one of the very famous algorithms in this area.

6.4 Indicator-based evolutionary algorithm (IBEA)

In this section, a local search algorithm for multi-objective optimization will be presented. Indicator-based evolutionary algorithm or IBEA was proposed in [92]. IBEA is an algorithm which can be combined with any indicator. The indicator is used to measure the performance of the solution. For IBEA, the users can choose the preference indicator by themselves. Moreover, it does not need the extra diversity preservation mechanism, e.g. the fitness sharing. In [92], a binary indicator was presented; it was used in the selection process. The binary indicators are used to compare the quality of two Pareto set approximations relatively to each other. The presented indicators are I_{ϵ^+}-indicator and I_{HD}-indicator. For instance, the I_{HD}-indicator, which is based

on the hypervolume concept, is given by

$$I_{HD}(A,B) = \begin{cases} I_H(B) - I_H(A) & \text{if } \forall x_2 \in B, \exists x_1 \in A : x_2 \prec x_1, \\ I_H(A+B) - I_H(A) & \text{otherwise} \end{cases}$$

where $I_H(A)$ is the hypervolume of the objective space dominated by A, $I_H(B)$ is the hypervolume of the objective space dominated by B, and $I_H(A+B)$ is the hypervolume of the objective space dominated by A and B [89]. Hence, $I_{HD}(A,B)$ is the volume of the objective space which is dominated by B but not by A in respect to a predefined reference point. The I_{HD}-indicator of the problem which needs to maximize $f_1(x)$ and minimize $f_2(x)$ is shown in Figure 1.10.

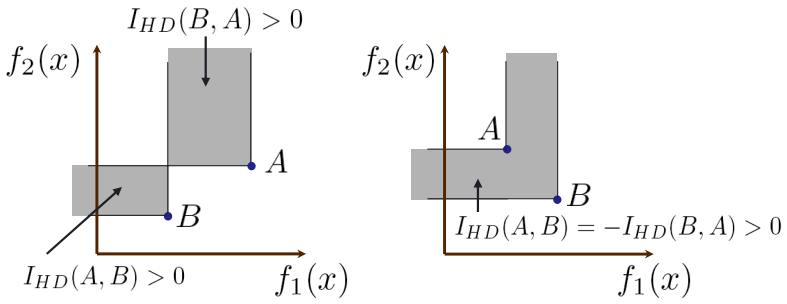


Figure 1.10: The I_{HD}-indicator of the problem which needs to maximize $f_1(x)$ and minimize $f_2(x)$.

The indicator can be integrated in the multi-objective evolutionary algorithm by using a fitness assignment. One simple possibility to assign the fitness for each individual is to sum up the indicator values for each population member with respect to the rest of the population as $F'(x^1) = \sum_{x^2 \in P \setminus \{x^1\}} I(\{x^2\}, \{x^1\})$, which is to be maximized. However, the basic IBEA proposed to use a slightly different scheme which is defined as

$$F(x^1) = \sum_{x^2 \in P \setminus \{x^1\}} -e^{-I(\{x^2\},\{x^1\})/\kappa}$$

where parameter κ is a scaling factor, which needs to be greater than 0, depending on I and the underlying problem. The main procedure of basic IBEA has the steps as in Algorithm 3.

Moreover, there is an adaptive IBEA, which improve the robustness. It scales both the objective values and the indicator values. The objectives values lie in the interval $[0,1]$. They are scaled by using the upper bound and lower bound of each objective. Because of the objective scaling, a reference

Algorithm 3 Procedure of IBEA

Step 1: Initialization
Set $t = 0$.
Generate randomly an initial population P_0 with size N.

Step 2: Fitness calculation
Calculate fitness values of individuals in P_t.

Step 3: Environmental selection
repeat

Step 3.1: Choose an individual $x^* \in P_t$ which has the smallest fitness value.
Step 3.2: Remove x^* form the population.
Step 3.3: Update the fitness values of the remaining individuals.
until $|P_t| \leq N$.

Step 4: Termination
if Stopping criteria are satisfied **then**
return Nondominated solutions in P_t
Stop
end if

Step 5: Mating selection
Perform binary tournament selection with replacement on P_t in order to fill the mating pool $P_t^{'}$.

Step 6: Crossover and mutation
Apply crossover and mutation operators to the mating pool $P_t^{'}$ and add the offspring to P_t.
$t \leftarrow t + 1$
Go to Step 2

point for I_{HD}-indicator can be determined easier. The worst value of each objective, which is 0 or 1 can be chosen, depending on the considered problem. For the indicator value, the values lie in the interval $[-1, 1]$ for all points in the population. They are scaled by using c - parameter, which is the maximum absolute indicator value. Hence, the adaptive IBEA changes Step 2 and Step 3 of the basic IBEA. In Step 2, the objective values are scaled and these scaled values are used to calculate the indicator values. Then, c - parameter is calculated and it is included in the equation of fitness assignment. Thus, the fitness assignment equation becomes as follows:

$$F(x^1) = \sum_{x^2 \in P \setminus \{x^1\}} -e^{-I(\{x^2\},\{x^1\})/(c \cdot \kappa)}$$

In Step 3, the equation for updating the fitness values of the remaining individuals also has to include the c-parameter.

6.5 $\mathscr{S}$ *metric selection* evolutionary multi-objective optimization algorithm (SMS-EMOA)

$\mathscr{S}$ *metric selection* evolutionary multi-objective optimization algorithm or SMS-EMOA was proposed in [12]. SMS-EMOA applies the use of hypervolume measure (or $\mathscr{S}$ metric) and the idea to maximize the dominated hypervolume within the optimization process for using as selection criterion. The algorithm consists of two main strategies which are:

- a nondominated sorting borrowed from NSGA-II, which is used as a ranking criterion, and
- a selection operator based on the hypervolume measure

  The individual, which obtains the least value of hypervolume, is assigned to be the worst individual in the considered rank.

The main procedure of SMS-EMOA has the steps as in Algorithm 4.

In the step for selecting N best individuals, the nondominated sorting from NSGA-II is applied to partition the current population into v ranks $\mathscr{R}_1, \mathscr{R}_2, \ldots, \mathscr{R}_v$. Afterwards, the last rank ($\mathscr{R}_v$) are considered. The individual, which has the lowest $\Delta_{\mathscr{S}}(s, \mathscr{R}_v)$ value, is eliminated. The $\Delta_{\mathscr{S}}(s, \mathscr{R}_v)$ value is defined as

$$\Delta_{\mathscr{S}}(s, \mathscr{R}_v) := \mathscr{S}(\mathscr{R}_v) - \mathscr{S}(\mathscr{R}_v \setminus \{s\})$$

where $\mathscr{S}$ is the hypervolume. Figure 1.11 shows the area which becomes the $\Delta_{\mathscr{S}}(s, \mathscr{R}_v)$ value.

Algorithm 4 Procedure of SMS-EMOA

Step 1: Initialization
Set $t = 0$.
Generate an initial population P_0 with size N.

Step 2: Iteration
repeat

Step 2.1: Generate the offspring by using variation operators and include it into the current population P_t.
Step 2.2: Select N best individuals from the current population P_t and store these individuals into the next population P_{t+1}.
Step 2.3: Increment the generation counter $t = t + 1$
until Stopping criteria are satisfied

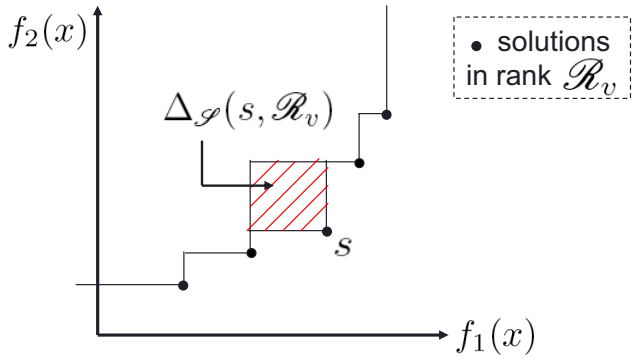


Figure 1.11: The area which becomes the $\Delta_{\mathscr{S}}(s, \mathscr{R}_v)$ value of the problem to maximize $f_1(x)$ and minimize $f_2(x)$.

Moreover, there is a modified selection procedure, which separates the selection criterion in two cases. The first case is used when the population can be classified in several nondominated ranks ($v > 1$). In this case, it selects the individual by considering the number of dominating points $d(s, P(t))$. It is the number of points from set $P(t)$ that dominate point s, which is given by

$$d(s, P(t)) := |\{y \in P(t) | y \prec x\}| .$$

The algorithm discards the individual with the highest $d(s, P(t))$ value among the solutions of the worst rank. For the second case, the $d(s, P(t))$ values of all individuals equal to zero or it means that all individuals are nondominated, the $\Delta_{\mathscr{S}}(s, \mathscr{R}_v)$ values are considered. The individual with the lowest $\Delta_{\mathscr{S}}(s, \mathscr{R}_v)$ value is eliminated as the non-modified algorithm.

6.6 Indicator-based multi-objective local search (IBMOLS)

Indicator-based multi-objective local search or IBMOLS is a simple and generic algorithm which was proposed in [7]. It borrows a part of the IBEA, which is a binary indicator using as the selection operator. The advantage of the binary indicator is no requirement of additional diversity preservation mechanisms. The binary indicator can compare two single solutions, or a solution against an entire population. Hence, it can be applied in the selection process of evolutionary algorithm in order to delete the worst solution or select the best solution. As with IBEA, the fitness assignment can be calculated from the binary indicator values. Hence, IBMOLS was proposed by combining the single objective iterated local search and indicator-based fitness assignment from IBEA. The procedure of baseline IBMOLS and the iterated local search have the steps as in Algorithms 5 and 6, respectively.

There are three parameters of IBMOLS that have to be defined. They are the population size, binary indicator, and population initialization. For the population size, a fixed one was suggested to use [7]. It can avoid the problem of the loss of diversity, in the case that only one nondominated solution is found, and slow down the convergence, in the case that the number of nondominated solutions grows up exponentially. For the binary indicator, several methods can be used depending on the considered problem. For the population initialization, it is managed into two cases. The initial population of the first iteration can be generated randomly or generated by using the useful data from the considered problem. The initial population of the next iterations can be generated randomly as for the first iteration or keeping information from the archived nondominated set of solutions. Hence, the users have to define these parameters depending on the considered multi-objective optimization problem.

As the steps of the algorithm, the neighborhood of the original IBMOLS is generated in a random order and each neighbor in the neighborhood is generated once. The generated neighbor is added to the population and the worst solution is removed, until the new solution is found. This algorithm uses the first improvement method of the local search for choosing the neighbor. However, we also apply IBMOLS to solve our problem. Some steps are modified, as it will be explained in Chapter 4.

7 Conclusion

In this chapter, the basic concepts of multi-objective optimization are described. As real-world optimization problems naturally involve many conflicting objectives, multi-objective optimization algorithms can be applied to solve such problems efficiently. When multi-objective optimization problems

Algorithm 5 Procedure of baseline IBMOLS

Step 1: Initialization
Generate an initial population P with size N.

Step 2: Archive set management
Store the nondominated solutions of the population P in an archive set A.

Step 3: Fitness assignment
For all individuals z in the initial population, calculate the fitness values from the indicators, i.e., $Fit(z) = I(P \setminus \{z\}, z)$.

Step 4: Local search step
for all individuals z in population P **do**

repeat
Step 4.1: Generate a neighbor x^* in the neighborhood X.
Step 4.2: Calculate the fitness values, $Fit(x^*) = I(P, x^*)$.
Step 4.3: Add neighbor x^* in the population P.
Step 4.4: Update the fitness values of all individuals z in the population P (except x^*), $Fit(z) += I(x^*, z)$.
Step 4.5: Select the worst individual w from the population P.
Step 4.6: Remove w from the population P.
Step 4.7: Update the fitness values of all individuals z in the population P, $Fit(z) -= I(w, z)$.
until All neighborhoods are explored or $w \neq x^*$.
end for

Step 5: Archive set update
Store the nondominated solutions of $A \cup P$ in the archive set A.

Step 6: Termination
if A does not change **then**

return A
else
Perform another local search step.
end if

Algorithm 6 Procedure of iterated local search

Step 1: Initialization
Create the approximate Pareto set $PO = \emptyset$.

Step 2: Iteration
while Stopping criteria not reached **do**
Step 2.1: Run the baseline IBMOLS and obtain the archive set A.
Step 2.2: $PO \leftarrow$ nondominated solutions of $PO \cup A$.
end while

return PO.

are solved, we hope to obtain only one best solution. Actually, a set of solutions are found, and it is difficult to decide that which one is better than the others, because of the contradictory objective functions. Hence, the dominance relation in the Pareto sense is used to decide the interest of the solutions by obtaining the nondominated solutions on the tradeoff surface. However, the search space can be too large and too complex to be solved by exact methods, therefore some of the metaheuristics are presented. Moreover, many evolutionary algorithms for multi-objective optimization can solve the problems efficiently in many applications. Some of them are explained in this chapter.

In the next chapter, an application which concerns a scheduling problem of Earth observing satellites, considered as a multi-objective optimization problem, will be presented.

Multi-user observation scheduling for Earth observing satellites

2

1 Introduction

Multi-objective optimization is used to model many real-world problems, especially, in aeronautical and aerospace applications, such as aerodynamics, structure, propulsion, acoustics, manufacturing, and economics [4]. Here, we consider an observation scheduling problem of Earth observing satellites. In Section 2, the mission and properties of the Earth observing satellites are explained. In this work, we study two types of Earth observing satellites, which are a non-agile and agile Earth observing satellites. The overview of the scheduling problem for both types of Earth observing satellites is described in Section 3. This section also explains the difficulty for solving the observation scheduling problem of agile satellite, when compared to non-agile satellite. In Section 4, the observation scheduling problem of an agile Earth observing satellite is considered as a multi-objective optimization problem and the model of the problem is presented. In the last section, the problem instances are described in detail; they will serve for the computational experiments (Chapter 5).

2 Earth observing satellites

Earth observing satellites (EOSs) have the mission to acquire photographs of the Earth surface in order to satisfy the requirements from users. The Earth observing satellites can acquire photographs, during moving along their orbits. They spend a period of several days for performing a cycle of orbits.

The whole area of the Earth is viewed, when the satellites complete a full cycle [47]. The Earth observing satellites carry the different instruments depending on their usages, e.g. optical camera or infrared cameras. Most of them operate at low altitudes. Hence, when they move over the visible areas of the required photographs, the photographs can be captured as in Figure 2.1. Then, the satellites will try to transfer the data of the acquired images directly to the ground station center after acquiring, if it is possible. Otherwise, the data are stored in the on-board memory with a limitation size, until the satellites are in the possible transferring range to the ground station center. There are many types of Earth observing satellites, which are used to acquire the images in order to support governments, research institutes, commercials, and militaries. Moreover, the satellites can be used to monitor and forecast the weather, observe the environment, map locations, manage natural resources, etc.

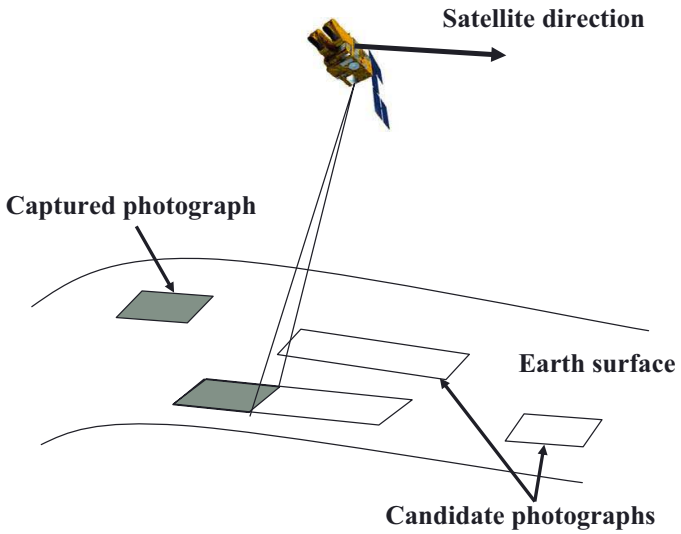


Figure 2.1: The satellite captures the photographs [64].

The types of satellites which are considered in this work are classified depending on the physical properties of the satellites. They are non-agile and agile satellites. These satellites are equipped with high resolution optical or/and infrared instruments.

2.1 Non-agile satellite

Non-agile satellite is a kind of Earth observing satellites which has only one degree of freedom for acquiring photographs such as SPOT satellite. The SPOT is a family of Earth optical observing satellites, which was initiated

by the CNES (Centre national d'études spatiales or French Center for Spatial Studies) in the 1970s. Several versions of SPOT have been developed and launched until now. The first SPOT (SPOT 1) was launched in 1986 and the launch of the latest one (SPOT 7) is planned in 2014. However, SPOT 5 is a non-agile Earth observing satellite, which is studied in this work and its properties are explained in this part. SPOT 5 is equipped with three high resolution visible imaging instruments, which are fixed on-board on the satellite. Each instrument is installed in different positions, which are front, middle, and rear. In front of each camera, there is a mobile mirror, which is attached. This mirror stays fixed during taking the photographs, but it turns around the roll axis during the transition from one acquired photograph to the other one, in order to prepare for taking the next photograph [61]. Two types of photographs can be taken from SPOT 5. They are mono photograph and stereo photograph. For obtaining the mono photographs from this satellite, one image, which is acquired from one of the three cameras, is needed. For obtaining the stereo photograph, it needs one acquired image from the front camera and another one from the rear camera [10]. As previously presented, the mobile mirror of each camera on SPOT 5 can turn around only the roll axis, therefore the starting time for acquiring each image is fixed. It is the time that the satellite moves over the beginning position of the acquired area. Hence, for solving the observation scheduling problem of SPOT 5, the set of the feasible sequences to acquire the images can be pre-computed. Then, each image will be assigned to the cameras [61].

2.2 Agile satellite

Another type of Earth observing satellites is agile satellite. The agile satellite, which is considered in this work, is equipped with only one fixed on-board camera, but the whole satellite can turn around three axes: roll, pitch, and yaw. The satellite uses an orbit control system to be able to move in these three degrees of freedom [59]. This property allows the satellite to take the required photographs by using only one camera. Each request can also be of two types: mono and stereo. Each area is taken only once for a mono request, whereas for a stereo request, each area must be acquired twice in the same direction but from different angles. An example of an agile satellite is PLEIADES, which was also developed by CNES. As the three axes that the agile satellite can move, the starting time for taking each image is not fixed, but it must be in a given time interval which is called time window. Because of this, an agile satellite has an important advantage when compared to a non-agile satellite. On the one hand, this gives agile satellite better efficiency of the whole system. On the other hand, the problem of selecting and scheduling the candidate images, is more difficult to solve, because the search space of

the scheduling problem for the agile satellite is larger than the search space for the non-agile satellite [61].

3 Observation scheduling of Earth observing satellites

As previously presented, the mission of Earth observing satellites is to obtain photographs of the Earth surface depending on requests from users. The mission management which concerns the observation scheduling problem is considered in this work and thus it is presented in this section. The satellite management process starts when several users order requests to a ground station center. After that, the ground station center has to manage the requests by selecting and scheduling the candidate images, according to some limitations of the satellite, before the obtained sequence is transmitted. If the requests are acquired by the satellite, they can give profits and gains. Hence, one way for obtaining the highest profit is that the ground station center tries to assign as many requests as possible to the satellite. But in the real case, the satellite usually cannot acquire all requests in only one revolution, because of several reasons. For example, the number of requests may exceed the satellite capacity, or several required areas must be acquired during the same period. Each request is represented as a candidate photograph in the observation management problem, which is considered in this work. Therefore, the considered problem is to select and schedule the feasible subset of candidate photographs for satisfying some objectives (e.g. maximize the total profit, maximize the number of acquired requests, etc.) and subject to the imperative physical constraints of the considered Earth observation satellite.

Many researchers studied scheduling problems for Earth observing satellites and have proposed several strategies and several algorithms to solve the problems. For examples, in [40] and [41], a project of NASA was studied. It was an oversubscription scheduling problem which considered the requests from users and the size of requests exceeded the satellite capacity. In the project, the users represented financial investors. The objective function was computed by the weighted sum of the observation's priority and the total spending time. Genetic algorithm, hill climbing, and simulated annealing were tested and compared. Simulated annealing obtained the best results. In [16], the allocation problem was studied. It needed to allocate a finite set of resources to a set of agents with fairness and efficiency. The experiments had done on the satellite application. Two constraint programming algorithms were proposed. In [88], the selection problem of multiple imaging frames for Earth observing satellite was considered. The objective of this problem is to assign each imaging frame in a given time slot and maximize the reward. The

problem was modeled as a directed acyclic graph. The nodes represent the candidate image frames and the directed edges imply the reachable successive imaging frames, which were defined by the constraints of Earth observing satellite. The path, which obtained the maximum reward, was a result of the problem.

Furthermore, the satellite constellation scheduling problem was also studied. In [52], this problem was decomposed into two subproblems: the task assignment and single satellite scheduling. The former considers the assignment of a task to a satellite. The latter schedules the assigned tasks to a given satellite. The objective was to maximize the number of completed images under the satellite operation constraints. An ant colony optimization algorithm and a simulated annealing heuristic were used to solve this problem. In [85], they mainly considered in the download scheduling mechanism that the acquired images had to be transferred to the ground station. A priority-based heuristic to avoid conflicts was proposed. In [14], planning and scheduling algorithms were studied for the COSMO-SkyMed constellation. It was one of the joint development satellite constellations between France and Italy. This constellation consisted of four satellites. Each satellite was phased at 90 degrees difference from the other one. The four satellites synchronized the operations, which needed to acquire the required images and transmit the data files to a set of ground stations. The objective was to maximize the number of images taken and transmitted.

3.1 Observation scheduling of non-agile Earth observing satellites

The non-agile satellite, which is studied in this work, is SPOT 5. The scheduling problem for SPOT 5 concerns daily photograph scheduling. The candidate photographs of the next day need to be managed before submission to the satellite. The main objective of this problem is to select a subset of photographs from a candidate set and to schedule it for obtaining a sequence, which uses the satellite efficiently. Hence, the selected set of photographs and their schedule should give the highest profit under operation constraints. The starting time for acquiring each photograph of SPOT 5 cannot be changed, according to its properties that SPOT 5 can move in only one degree of freedom. Thus, the acquiring starting time is the exact time when the satellite flies over the beginning position of the desired area.

SPOT 5 instances were proposed in [10]. This set of instances consists of 20 instances which are divided in two groups. The first group is the data from the single satellite revolution and do not consider a recording capacity constraint. The other group involves several satellite revolutions and also

includes a recording capacity constraint. In both instance groups, the common considered constraints are:

1. Non-overlapping: A satellite cannot acquire more than one photograph in the same period of time.

2. Minimum transition time: A transition time is a necessary time to move the camera from the ending point of the previous photograph to the beginning point of the next photograph.

3. Limitations of the instantaneous data flow through the satellite telemetry: The satellite will try to transfer the data of the acquired photographs directly to the ground station after acquiring, if it is possible. Otherwise, the data are stored in the on-board memory with a limitation size, until the satellite is in the possible transferring range to the ground station.

The data description of each instance was explained in [10]. Each instance consists of:

- Set of requested photographs $S = \{s_1, s_2, \ldots, s_n\}$
- Set of cameras $C = C_1 \cup C_2 = \{1, 2, 3, 13\}$
  - $C_1 = \{1, 2, 3\}$ is the set of cameras for mono photographs
  - $C_2 = \{13\}$ is the set of cameras for stereo photographs
- $\forall s_j \in S$, $p_j \in \mathbb{N}$ is the profit if photograph s_j is selected
- $\forall s_j \in S$, $\alpha_j \in \{1, 2, 3\}$ is the number of possible cameras for taking photograph s_j
- $\forall s_j \in S$, $PC_j \subseteq C$ is the set of possible cameras for taking photograph s_j
- $\forall s_j \in S$, $\exists c \in PC_j$, $w_{cj} \in \mathbb{N}$ is the recording consumption of photograph s_j be taken by camera c (only for instances in the second group)
- Set of binary and ternary constraints $U = \{u_1, u_2, \ldots, u_m\}$
- $\forall u_k \in U$, $\beta_k \in \{2, 3\}$ is the number of (s_j, c) that are forbidden to be assigned in the same solution,
  - if $\beta_k = 2$, it means that at most one of (s_{j_1}, c_1), (s_{j_2}, c_2) can be selected (*Constraint2*)

– if $\beta_k = 3$, it means that at most one or two of (s_{j_1}, c_1), (s_{j_2}, c_2), (s_{j_3}, c_3) can be selected (*Constraint3_1* or *Constraint3_2*, respectively)

where (s_j, c) is an associated pair which represents the photograph s_j to be taken by camera c

- The maximum recording capacity on board W_{max} (only for instances in the second group)

Several formulations were proposed for the SPOT 5 observation scheduling problem. The formulation as a knapsack model was proposed in [80]. In [35], two different formulations were presented. The first one is a formulation which uses mathematical programming. The second one is a formulation from graph theory model. In [69], another formulation based on valid inequalities arising in node packing and 3-regular independence system polyhedra was proposed.

In this work, we will explain in detail the formulation based on a knapsack model. Each mono photograph s_j in S can be acquired by one of the three cameras denoted $1, 2$, or 3. Hence, three associated pair $(s_j, 1)$, $(s_j, 2)$, and $(s_j, 3)$ are considered. Similarly, an associated pair $(s_j, 13)$ is considered for each stereo photograph s_j in S, since the stereo photograph needs the two cameras (front and rear) for acquisition. Hence, the number of $3 \cdot n_{mono} + n_{stereo}$ associated pairs, which are considered in each instance, is equal to m.

$$m = 3 \cdot n_{mono} + n_{stereo}$$

where n_{mono} is the number of mono candidate photographs and n_{stereo} is the number of stereo candidate photographs. Then, the photograph schedule corresponds to a binary vector, which elements represent all associated pairs (s_j, c). The binary vector is

$$x = (x_1, x_2, \ldots, x_m)$$

where $x_i = 1$, if its associated pair is present in the schedule, otherwise, $x_i = 0$.

Each profit p_j of all candidate photographs is transformed to a set of profits which correspond to all associated pairs. It is represented by

$$px = (px_1, px_2, \ldots, px_m)\,.$$

Similarly, each recording consumption w_{cj} is transformed to a set of recording consumptions which correspond to all associated pairs (s_j, c). It is represented by

$$wx = (wx_1, wx_2, \ldots, wx_m)\,.$$

The optimization problem of these instances is to maximize the total profit value of the schedule x and subject to all constraints, which is given by

$$\begin{aligned}
\text{maximize} \quad & f(x) = \sum_{1 \le i \le m} x_i \cdot px_i \\
\text{subject to} \quad & \sum_{1 \le i \le m} wx_i \cdot x_i \le W_{max}, \\
& \forall (x_{i_1}, x_{i_2}) \in \textit{Constraint2}, x_{i_1} + x_{i_2} \le 1, \\
& \forall (x_{i_1}, x_{i_2}, x_{i_3}) \in \textit{Constraint3_1}, x_{i_1} + x_{i_2} + x_{i_3} \le 1, \text{and} \\
& \forall (x_{i_1}, x_{i_2}, x_{i_3}) \in \textit{Constraint3_2}, x_{i_1} + x_{i_2} + x_{i_3} \le 2.
\end{aligned}$$

$Constraint2$ involves the non-overlapping, the minimal transition time, and limitations on instantaneous data flow. It is expressed by the relation of two pairs. It forbids to select these two pairs simultaneously for assigning in the sequence. $Constraint3\_1$ represents a constraint, which forbids to select more than one pair from the same mono photograph. $Constraint3\_2$ involves limitations on instantaneous data flow, which cannot be expressed in form of $Constraint2$.

Several algorithms were applied to solve this problem. For example, a Tabu search algorithm was used in [80]. The results were separated in two groups depending on the set of instances. For the first group of instances, the proposed algorithm found easily all optimal solutions. For the second group of instances, it improved some best-known solutions and required less computation times. In [81], an upper bound of the solution for each instance was proposed. The upper bounds were obtained by applying an original partition-based approach. A Tabu search algorithm was used to determine optimized partitions.

Moreover, the SPOT 5 scheduling problem was considered as a multi-objective optimization problem in [64]. A genetic algorithm was proposed to solve this problem. Two objective functions were considered. They were the maximization of the total profit and maximization of the number of acquired photographs. Each gene of chromosome represented the candidate image. The values of genes were encoded differently from the $0-1$ knapsack model formulation as the previous one. They were encoded by using the integer $0, 1, 2, 3$, or 13. The values represented the cameras which were used to acquire the associated candidate images. If the gene value was equal to $1, 2$, or 3, it meant that the associated candidate image was a part of the mono photograph and was assigned to be taken by front, middle, or rear, respectively. If the gene value was equal to 13, it meant that the associated candidate image was a part of the stereo photograph and was assigned to be taken by front and rear cameras. If the gene value was equal to 0, it meant that the associated candidate image was not assigned to be taken from the satellite.

As previously presented, the starting time for taking the candidate photographs of the non-agile satellites cannot be changed and the set of feasible taken photographs can be pre-computed. In [61], the authors noticed that the management problem for the non-agile satellite is only a selection problem, not a true scheduling problem. Therefore, the scheduling problem for the agile Earth observing satellite, which has larger search space and is more difficult to be solved, will be mainly consider in this work.

3.2 Observation scheduling of agile Earth observing satellites

The observation management for the agile observing satellite has the same goal as for non-agile satellite, which is to select and schedule the candidate photographs for assigning an efficient sequence to the satellite. Generally, the obtained sequence should give the highest profit and satisfy the satellite constraints.

The request management of agile satellite must consider the request type and also request shape. Each request can be of two types: mono or stereo. Each area is taken only once for mono requests, whereas for stereo requests, each area must be acquired twice in the same direction but from different angles. There are two possible shapes: spot or polygon. The spot is a small circular area with a radius of less than 10 km. The polygon is a polygonal area ranging from 20 to 100 km. Both shapes have to be managed by transforming the requests into several rectangular shapes called strips. Each strip can be taken once at a time by the camera on the satellite. A spot is considered as a single strip. For each polygon, the area is too big for being taken only once, it is decomposed into several strips. All strips have the same width but variable lengths. The example of request shapes and the example of the order for taking the strips after management are illustrated in Figure 2.2. There are two possible directions which can acquire each strip. Both directions are parallel to the length of each strip, but in the opposite directions as shown in Figure 2.3. Among two of them, only one acquired direction can be selected. The strip, which is associated with one possible acquired direction, is called an acquisition. Thus, each strip consists of two possible acquisitions. In the observation scheduling problem for agile Earth observing satellite, some acquisitions will be selected and scheduled in order to obtain the sequence, which is the solution of this problem.

The starting time for acquiring each acquisition is not fixed, but it has to be in a given time interval, which is called a time window. The possible starting time for taking each acquisition can be computed, depending on the acquired direction, the earliest and latest visible time of the two extremities of the strip, and the acquired duration time of the strip. Moreover, the adjacent selected acquisitions must also respect a sufficient transition time. The sufficient tran-

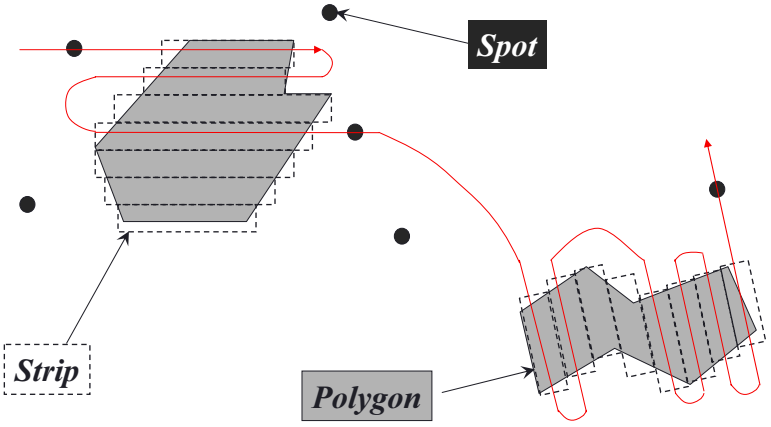


Figure 2.2: The example of both request's shapes and the order for taking the strips after management [61].

sition time is the necessary time to move the camera from the ending point of previous acquisition to the beginning point of the next acquisition.

Hence, the selecting and scheduling problem of the acquisitions is considered in this work. The set of the selected acquisitions are ordered to become the sequence for being transmitted to the satellite. The obtained sequence has to satisfy the imperative constraints of agile Earth observing satellite. The first constraint is the time window, where the starting time of the selected acquisition must be in the given time interval. The second one is the sufficient transition time, where the camera of the satellite has enough time to be moved from the ending point of the previous acquisition to the starting point of the next one. For the third constraint, at most one of the two possible directions of each strip is selected to be acquired. The fourth one is the stereo constraint for the stereo photographs, if one of the twin acquisitions are selected, the other one should also be selected, where the twin acquisitions take the same strip, in the same direction, but different angles.

When some acquisitions are selected to be acquired by the satellite, they give the profit or gain. Thus, for the observation scheduling problem, the objective, which is total profit maximization, is considered. The total profit is computed depending on the acquired area of each request. The profit of each acquired request can be computed by using a piecewise linear function of gain. This function is associated with a fraction of the acquired useful area and the whole area of each request, as illustrated in Figure 2.4. The request, that more area of them is taken, also gives more profit.

There are several studies on agile Earth observing satellites. For example, a combination of genetic algorithm and simulated annealing, was proposed to solve this problem in [62]. The performance of the proposed algorithm was

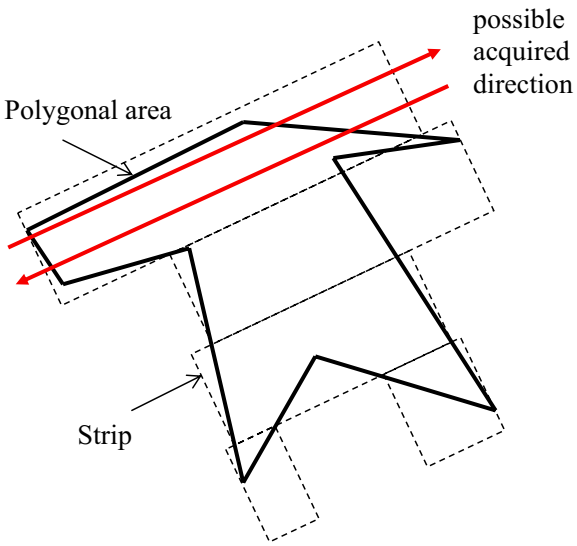


Figure 2.3: A polygonal area is decomposed into several strips; each strip can be acquired according to two possible directions [61].

compared with the simulated annealing alone. In [61], four methods consisting of a greedy algorithm, a dynamic programming algorithm, a constraint programming algorithm, and a local search method were applied in order to solve a simplified version of the scheduling problem for agile Earth observing satellites.

The ROADEF 2003 challenge was about the management problem of agile Earth observing satellite mission (see http://challenge.roadef.org/2003/en/). The challenge deals with a scheduling, which optimize an objective (total profit maximization) and also satisfy the imperative constraints of the satellite. The data's description and the optimization criterion were explained in [82]. The criterion of this challenge is to maximize a gain criterion, which is the sum of requests' gains that are associated with the complete or partial acquisition of each request. The winner of this challenge used an algorithm based on simulated annealing for solving the scheduling problem [57]. The decision to accept move depended on a random chance with the percentage loss of the achievement, when moving from the current solution to the considered neighbor. Two types of moves were used to define the neighborhood in this proposed algorithm. The second prize winner proposed an algorithm based on tabu search [22]. The authors noticed that the scheduling problem of agile satellite was similar to the selective traveling salesman problem. However, there were some different features, e.g. the non-linear objective function, the linking constraints, and the time windows. Therefore, they adapted the unified tabu search algorithm [21], which was developed for

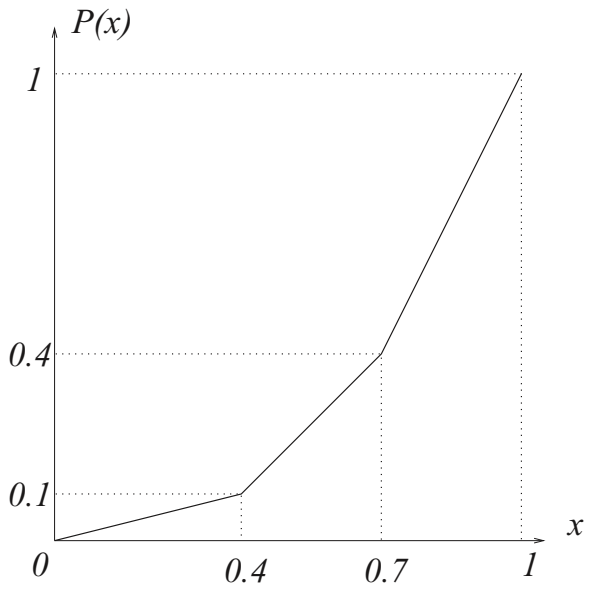


Figure 2.4: Piecewise linear function of gain $P(x)$ depending on the effective ratio x of acquired area [82].

the vehicle routing problem with time windows. The relaxed constraints and self-adjusted parameters were applied to penalize infeasible solutions. Six types of moves were considered in order to define the neighborhood. In [47], a tabu search algorithm hybridized with a systematic search was applied to solve this problem. The authors used a formulation of constrained optimization problem and a convex evaluation function. Moreover, the upper bounds were computed by relaxing the constraints and linearization of the objective function. The proposed algorithm was also verified on the instances from the ROADEF 2003 challenge.

These works considered the scheduling problem for the agile Earth observing satellite as a mono objective optimization problem. One objective function, which was the total profit maximization, was treated. In our work, the scheduling problem is considered in the case that the requests are required from multiple users. Hence, it is considered as a multi-objective optimization problem, which needs to optimize not only one objective. The original instances of ROADEF 2003 challenge, which were described in [82], will be modified for testing the proposed algorithms for solving the problem. The description of all data for the multi-objective optimization problem will be explained in the next section.

4 Multi-objective scheduling of agile Earth observing satellites

The scheduling problem of agile Earth observing satellites, where the requirements emanate from several different users, is considered in this work. We need to optimize two objective functions, which are to maximize a total profit and simultaneously ensure the fairness of resource sharing for all users. Thus, this problem is modeled as a multi-objective optimization problem.

Some researchers studied multi-objective optimization problems for space applications [4]. For example, the scheduling problem of SPOT satellites considering three objective functions was studied in [36]. The three objective functions were the maximization of number of shots for satisfying the users, the maximization of the total profit according to the request priority, and the minimization of the satellite use. The last objective was contradictory with the first two objectives and the number of on/off times of the instruments was used for measuring the satellite use. The authors modeled the problem as a selection of a satisfactory efficient path in a graph without circuit. The path selection process was separated into two stages. The first stage was the generation of efficient paths and the second one was the selection of a satisfactory path by using the interactive multiple criteria procedure. In [84], the scheduling problem of Earth observing satellites, which was modeled by using the multi-objective optimization, was studied. Strength Pareto Evolutionary Algorithm 2 (SPEA 2) was applied to solve this problem. Two objective functions were the maximization of the importance of acquired requests and the minimization of the resource consumption. Moreover, the satellites constraints had to be satisfied. The genetic operators were designed in order to avoid generating a huge number of infeasible solutions in the process. The problem coding step, crossover and mutation operators were developed for the proposed approach.

In [13], the management problem with multiple satellites, multiple orbits, and multiple users, was considered. A tabu search was applied for selecting and scheduling the requests from the users under the operational constraints. Three phases of allocation process were considered. The first phase was the selection of priority requests. The second phase solved the optimization problem in order to select the requests from the subset of users. The last one also solved the optimization problem for sharing the remaining capacities of the satellites between all users. The considered objective function was the weighted sum of the normalized utilities of the users. It was adapted from the order weighted average in [86]. This operator for aggregation was used in order to ensure the fairness of the solution. The utility of each user was defined as the sum of obtained profits according to the acquired requests of that user. Then, column generation procedure was applied to compute the upper bound

for evaluating the quality of the solutions. The developed Tabu search was tested with data instances from CNES.

As previously presented, several related works considered the requests, which are required from only one user. An objective, which is the total profit maximization, is sufficient to be optimized to find the solution. But in the case that the requests are required from several users, other objectives should also be considered, especially, the fair sharing of the satellite resources for all users. There is some literature which considered to ensure the fairness among users. As in [13], the fairness was taken into account, but it was not considered as an objective function. In [58], the importance of fair sharing was illustrated. They presented that normally, a huge financial investment had to be used for space missions (e.g. the satellite mission) and several entities had to pay for this. Thus, they needed the sufficient uses of the space system according to their investments. Hence, the sharing rules between different entities must be satisfied. Moreover, the selection problem for the agile Earth observing satellites concerning multiple end-users was considered and sharing principles were adopted to select the candidates based on the utility levels. In [8] and [60], the use of two objective functions related to fairness and efficiency was proposed. Three ways were discussed for solving this sharing problem: the first one gives priority to fairness, the second one to efficiency, and the third one computes a set of trade-offs to help a human to make decisions. For the multi-criteria methods, instead of building a complete set of nondominated solutions, the authors only searched for a solution, which is close to the line with a specified slope on the objective function plane. It is a compromise solution between two other solutions, which are obtained by giving priority on an objective.

Therefore, in this work, the management of the candidate requests of an agile Earth observing satellite is placed in a multi-objective optimization framework. The management process needs to select and schedule the possible acquisitions, and then transmit the sequence of the selected acquisitions to the satellite. The requests, which are ordered from multiple users, are considered. The obtained sequence has to maximize the total profit and simultaneously fairly share the satellite's uses among the users. The minimization of the maximum profit difference between users is used to ensure the sharing fairness. Among the set of solutions, which contains the sequences of the selected acquisitions, the decision-maker can choose a preferred sequence for transmitting to the satellite. The instances of ROADEF 2003 challenge, which simplified the problem and explained in [82], are modified by including a specific user for each request. The data description of one instance, decision variables, considered constraints, and objective function computation are presented below.

5 Instances description

In this section, the instances, which are modified from the ROADEF 2003 challenge instances, will be described in detail. These instances will be used in the experiments, for testing the performance of the proposed algorithms in our work (see Chapter 5.

5.1 Data

- a set of requests $Req = \{1, 2, \ldots, i, \ldots, n_r\}$
- a set of strips $Strip = \{1, 2, \ldots, j, \ldots, n_s\}$
- a set of possible strip acquisitions $Acq = \{1, 2, \ldots, k, \ldots, n_a\}$, where $n_a = 2n_s$

  Note: Since one strip can be taken from two directions, the size of possible strip acquisitions set is then twice the size of strips set.

  - $k = 2j - 1$; the first direction that can take strip j
  - $k = 2j$; the second direction that can take strip j.

For each request i, $1 \leq i \leq n_r$:

- $U[i]$ is the user which orders request i
- $G[i]$ is the gain of request i for a complete acquisition (in $/km^2$)
- $S[i]$ is the surface area of request i (in km^2)
- $T[i]$ is the type of request i; target or polygon
- $St[i]$ is the mono/stereo characteristic of request i: 0 for mono and 1 for stereo.

For each strip j, $1 \leq j \leq n_s$:

- $R[j]$ is the request from which strip j was split
- $Tw[j]$ is the mono/stereo characteristic of strip j: 0 for mono and the index of its twin strip for stereo
- $Su[j]$ is the useful surface of strip j (in km^2)
- $Du[j]$ is the duration time for taking strip j (in seconds)
- The positions in the setting coordinates of the two ends of strip j are:

  - $X[j,0]$ and $Y[j,0]$ for the coordinate values of the end 0 position
  - $X[j,1]$ and $Y[j,1]$ for the coordinate values of the end 1 position.

  Note: The two possible directions for taking each strip are from end 0 to end 1 (first direction) and from end 1 to end 0 (second direction).

- The visibility times (in seconds) are:
  - $T_e[j,0]$ for the earliest visibility time of end 0 of strip j
  - $T_l[j,0]$ for the latest visibility time of end 0 of strip j
  - $T_e[j,1]$ for the earliest visibility time of end 1 of strip j
  - $T_l[j,1]$ for the latest visibility time of end 1 of strip j.

The data for the possible strip acquisitions have to be computed from the data of the associated strip.

For each possible strip acquisition k**,** $1 \leq k \leq n_a$**:**

- its earliest starting time is:
$$Tmin[k] = \max(Te[j,i], Te[j,i^{'}] - Du[j])$$
- its latest starting time is:
$$Tmax[k] = \min(Tl[j,i], Tl[j,i^{'}] - Du[j])$$
where $j = \lfloor (k+1)/2 \rfloor$, $i = (k+1) \bmod 2$, $i^{'} = k \bmod 2$ and the points $< j,i >$ and $< j,i^{'} >$ are the starting and ending points of the acquisition k.

For each pair of possible acquisition $k, k^{'}$**:**

- the distance between the ending point $< i,j >$ of acquisition k and the starting point $< i^{'}, j^{'} >$ of acquisition $k^{'}$ (in meters) is:
$$Di[k,k^{'}] = \sqrt{(X[j,i] - X[j^{'},i^{'}])^2 + (Y[j,i] - Y[j^{'},i^{'}])^2}$$
- the rotation (in radians) is:
$$Ro[k,k^{'}] = 2 \times arctan\left(\frac{Di[k,k^{'}]}{2Hs}\right)$$
where Hs is the satellite altitude in meters
- the upper bound of the time that is necessary to go from the ending point of acquisition k to the starting point of acquisition $k^{'}$ (in seconds) is:
$$Dt[k,k^{'}] = Dmin + \frac{Ro[k,k^{'}]}{Vr}$$
where $Dmin$ is an incompressible transition time (in seconds) and Vr is the maximum speed of rotation on itself of the satellite (in radian/second).

5.2 Decision variables

- a set of the selected acquisitions

  $acq\_sel = \{sa_1, sa_2, \ldots, sa_{id}, \ldots, sa_N\}$

- a set of the acquisition starting times which is related to the sequence of the selected acquisitions

  $start\_time = \{ta_1, ta_2, \ldots, ta_{id}, \ldots, ta_N\}$

  where N is the number of selected acquisitions

- a set of the request's profits

  $req\_profit = \{rp_1, rp_2, \ldots, rp_i, \ldots, rp_{n_r}\}$

  where n_r is the number of requests

- a set of the user's profits

  $user\_profit = \{up_1, up_2, \ldots, up_{iu}, \ldots, up_{n_u}\}$

  where n_u is the number of users.

5.3 Constraints

The imperative constraints must be satisfied for the feasible solutions.

For each selected acquisition sa_{id}, where $1 \leq id \leq N$:

- The time window is:

  $Tmin[k] \leq ta_{id} \leq Tmax[k]$

  where $sa_{id} = k$.

- The sufficient transition time is:

  $ta_{id} + Du[j] + Dt[k, k^{'}] \leq ta_{id+1}$

  where $sa_{id} = k$, $sa_{id+1} = k^{'}$, and j is the strip of acquisition k.

- At most one of the two directions is selected for each strip.

  In the sequence of the selected acquisitions acq_sel,

  - if one element (sa_{id}) equals to $2j - 1$, all others should not equal to $2j$, and
  - if one element equals to $2j$, all others should not equal to $2j - 1$.

- Stereo constraint

  For each pair of twin strips, when $Tw[j] \neq 0$, $Tw[j^{'}] \neq 0$, $Tw[j] = j^{'}$, $Tw[j^{'}] = j$.

  In the sequence of the selected acquisitions acq_sel,

- if one element equal to $2j-1$, one of all others should equal to $2j^{'}-1$
- if one element equal to $2j$, one of all others should equal to $2j^{'}$.

5.4 Computation of objective function values

Firstly, the set of the request's profits has to be computed as the steps below.

For each request i**,** $1 \leq i \leq n_r$**:**

- compute the taken useful surfaces of the request i ($SuR[i]$)

$$SuR[i] = \sum Su[\left\lfloor \frac{sa_{id}+1}{2} \right\rfloor]$$

for all id that $R[\lfloor \frac{sa_{id}+1}{2} \rfloor] = i$

- compute the fraction of the taken useful surface and whole surface

$$fr[i] = SuR[i]/((St[i]+1) * S[i])$$

- compute the complete profit of request i (100% of the area are taken)

$$gc[i] = G[i] * S[i] * (St[i]+1)$$

- compute the profit of request i that is associated with its partial acquisition

$$gr[i] = gc[i] * P(fr[i])$$

where function P is piecewise linear function on the interval $[0, 1]$

- store the profit of request i in the set of the request's profits (req_profit)

The profit of each user is computed and is stored in the set of the user's profits:

For each user, $1 \leq iu \leq n_u$**:**

- compute the profit of each user ($gu[iu]$)

$$gu[iu] = \sum gr[i]$$

for all i that $U[i] = iu$

- store the profit of user iu in the set of user's profits ($user\_profit$)

Finally, the objective functions values are computed:

- the total profit
$$\sum_{1 \leq i \leq n_r} rp_i$$
- the maximum profit difference between each pair of up_{iu}

In this work, the set of solutions on the approximate Pareto front will be obtained after solving this presented model by maximizing the first objective and minimizing the second objective. Moreover, the obtained solutions must satisfy the imperative constraints. Then, the decision maker can choose the preferred sequence from the set of solutions on the approximate Pareto front and transmits it to the satellite for operating the mission.

6 Conclusion

The details of the considered problem were explained in this chapter. It addresses the management of a space mission problem, which concerns Earth observing satellites. Two types of Earth observing satellites consisting of non-agile and agile satellites are presented. They have the general mission to obtain the photographs of the Earth surface for satisfying users' requirements. The non-agile satellite is equipped with three fixed on-board cameras, but can move according to only one degree of freedom. The agile satellite is equipped with only one fixed on-board camera, but the whole satellite can move in three degrees of freedom. The observation management problem of an agile satellite is considered in this work. As its capacity of photograph acquiring is more flexible, the underlying scheduling problem is more complicated to solve. The management means selecting and scheduling the possible candidate acquisitions to assign in the sequence. This work studied the problem where requests are ordered from several users. Thus, multiple objectives have to be considered simultaneously. The set of solutions in the approximate Pareto front, which satisfies the objective functions and respects the constraints, will be obtained. Then, the final decision must be made by the decision maker, but this part is not considered in this work. The best sequence of the management problem can be then transmitted to operate on the satellite.

In the two next chapters, two algorithms are proposed to solve this problem. The biased random key genetic algorithm and the indicator-based multi-objective local search are applied in Chapter 3 and 4, respectively. The computational results, which are obtained from both algorithms, are presented and discussed in Chapter 5.

Biased random key genetic algorithm

3

1 Introduction

In this work, the problem under study is the management problem of an agile Earth observing satellite. Requests from several users are considered. As described previously (see Chapter 2), the requests are transformed in several strips and each strip can be taken from two opposite directions. Thus, the possible acquisitions, which are the strips associated with acquiring directions, are selected and scheduled in order to obtain an efficient sequence. This chapter proposes a biased random key genetic algorithm (BRKGA) for solving the problem.

This chapter is organized as follows. Section 2 presents the details of BRKGA when it is applied to solve the selection and scheduling problem of Earth observations. The section exposes the composition of chromosomes and the encoding method, the solution improvement process, and the decoding strategies, which are used to decode the chromosomes to become the solutions. In Section 3, the BRKGA is proposed for solving the multi-user observation scheduling problem for an agile Earth observing satellite. It needs to optimize two objectives. Three selection methods of popular multi-objective evolutionary algorithms are applied in order to select the preferred solutions: nondominated sorting genetic algorithm II (NSGA-II), $\mathscr{S}$ *metric selection* evolutionary multi-objective optimization algorithm (SMS-EMOA), and indicator-based evolutionary algorithm (IBEA). Moreover, a hybrid decoding method is presented.

2 BRKGA applied to Earth observation scheduling problem

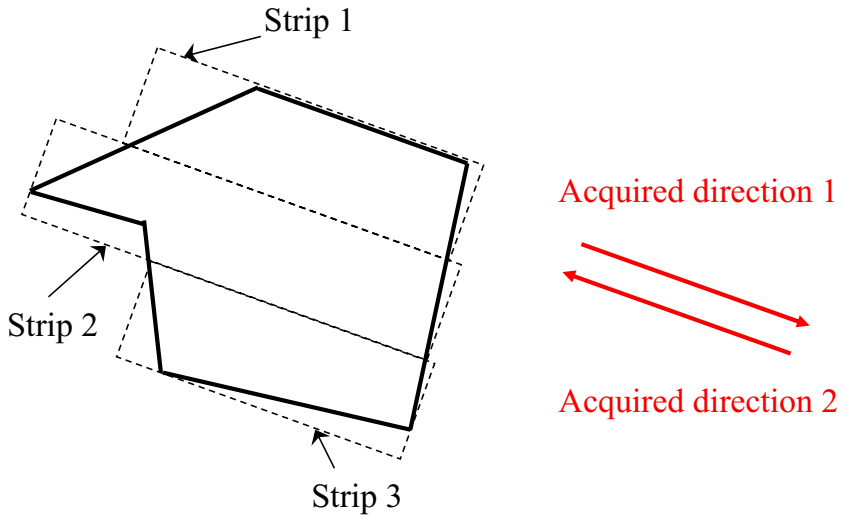
The biased random key genetic algorithm [44] (BRKGA), which combines genetic algorithm and the concept of random key, is used to solve the observation selecting and scheduling problem for an agile Earth observing satellite. There are important parts for adapting the BRKGA to solve the scheduling problem. The first part is how to define the chromosome representation. In this part, the encoding method, which is used to encode the decision variables of the problem into the gene values of the chromosome, is presented. The second part is the genetic algorithm process, which is used for improving the solutions found. The last part is the decoding method. It is applied to decode the chromosome from the genetic algorithm process in order to obtain the solution of the problem. Each of this part is detailed in the following successive sections of this chapter.

2.1 Random key chromosome and encoding method

The BRKGA is operated on several individuals in a population. Each individual contains a chromosome which represents a solution. The random key chromosome of BRKGA is formed by several genes. The gene's values are randomly generated in the interval $[0, 1]$. For the considered problem, each gene represents one acquisition. As described in the problem description in Chapter 2, the requests are ordered from the users and all requests have to be managed depending on their shape. The spot is considered as one strip and the polygon is decomposed into several strips. The acquisition, which is represented by the gene, is the strip associated with one possible acquired direction. Hence, the number of genes is equal to the number of possible acquisitions, which is twice the number of strips. In BRKGA, the random key chromosomes are operated by the genetic algorithm operators for generating the new population in each generation. Figure 3.1 illustrates example of acquisitions and associated random key chromosomes. These acquisitions come from three strips, which are decomposed from a polygonal request.

2.2 Genetic algorithm process

As already said in the introduction of BRKGA in Chapter 1, the population (its size is equal to p) is modified by using three operators: selection, crossover, and mutation. These operators generate three sets of chromosomes: the elite set, the crossover offspring set, and the mutant set. The order to generate each part of the population for the next generation is depicted in Figure 3.2.



Acquisitions

	Acquired direction 1	Acquired direction 2
Strip 1	Acquisition 1	Acquisition 2
Strip 2	Acquisition 3	Acquisition 4
Strip 3	Acquisition 5	Acquisition 6

Random key chromosome

Acquisition 1	Acquisition 2	Acquisition 3	Acquisition 4	Acquisition 5	Acquisition 6
0.6984	0.9939	0.6885	0.2509	0.4672	0.8293

Figure 3.1: Example of acquisitions and associated random key chromosomes.

The first generated part is the elite set. It consists of p_e chromosomes, which are selected among the preferred ones from the current population. In this part, if the considered problem needs to satisfy only one objective function, the chromosome selection can be done easily. However, for multi-objective problems, some selection methods for the multi-objective optimization should be applied to select the preferred chromosomes. These selection methods will be explained in detail in Section 3.1. The elite set is stored in the top position of the next population. The second generated part is the mutant set. The p_m chromosomes are generated randomly as the initial chromosomes. The mutant set is stored in the bottom position of the next population. The last generated part is the crossover offspring set. The crossover offspring set is

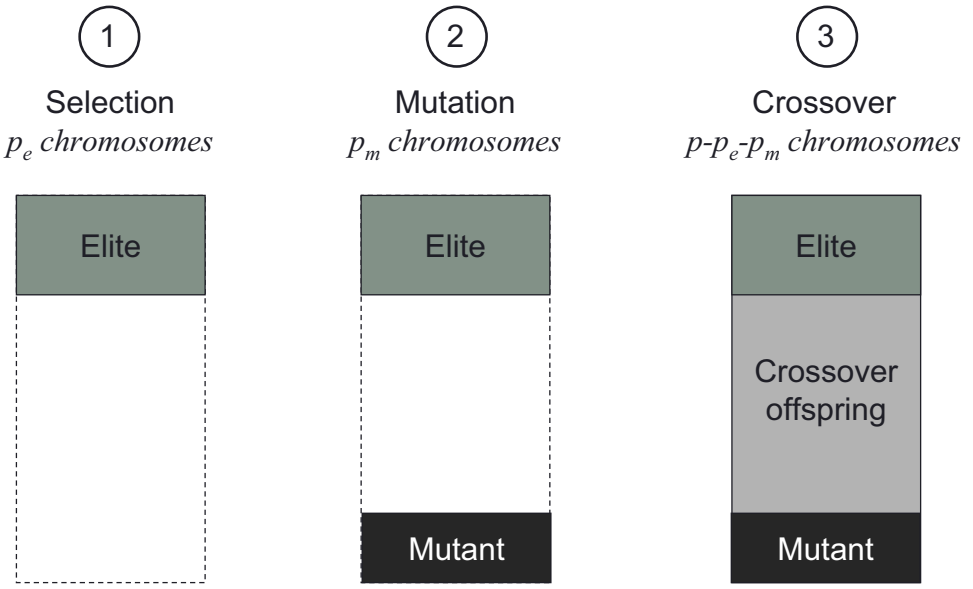


Figure 3.2: The order to generate each part of the population for the next generation.

stored in the middle position of the next population. Each offspring is built by using the crossover operator between a randomly selected chromosome from the elite set and another randomly selected chromosome from the whole current population. Each element in the offspring is obtained from the element of elite parent with the probability ρ_e. Otherwise, the element of offspring is copied from the non-elite parent. An example of BRKGA crossover operation is illustrated in Figure 3.3. The offspring is repeatedly generated until the remaining space of the next population is fulfilled.

2.3 Decoding strategies

The other important part of BRKGA is the decoding part. It is used to transform the random key chromosome to a solution for the considered problem. For the observation scheduling problem, each chromosome is decoded in order to obtain a sequence of the selected acquisitions. In this decoding step, the priority for selecting and scheduling each acquisition is considered depending on its associated gene value. The acquisition, which has the highest priority, is considered firstly to schedule in the sequence. Two main decoding methods are applied for solving this observation scheduling problem: a basic decoding and a decoding of gene value with ideal priority combination.

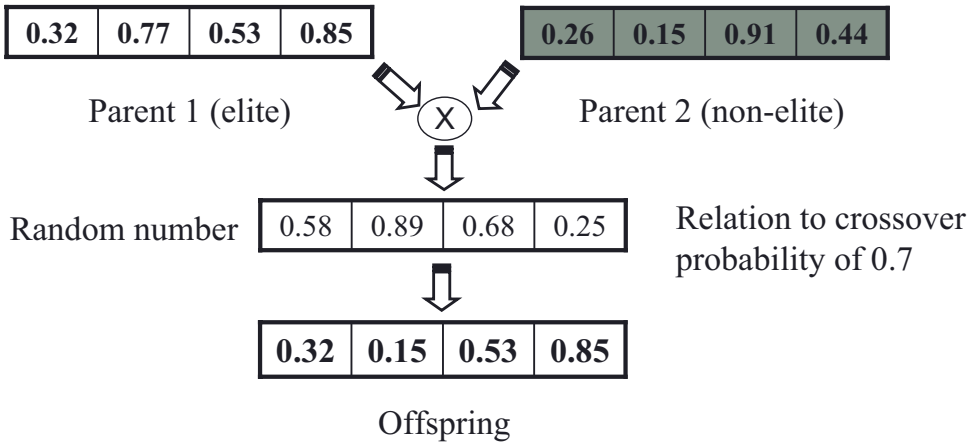


Figure 3.3: The example of BRKGA crossover operation [45].

2.3.1 Basic decoding

The priority of the basic decoding is defined by using directly the gene value. This decoding expression is given by

$$Priority_j = gene_j.$$

This basic decoding allows the highest priority for scheduling to the acquisition with the highest gene value. Thus, the priority of this decoding method depends only on the random key value.

2.3.2 Decoding of gene value with ideal priority combination

This decoding method is borrowed from [65]. It was applied for the resource-constrained project scheduling problem. This decoding defines the priority of each acquisition depending on two values: its associated gene value as in the basic decoding and its calculated ideal priority value. For the concept of ideal priority, the job, which has the earliest possible starting time, should be selected first and be scheduled in the beginning of the solution sequence. Hence, the ideal priority gives a higher priority to select and schedule the job with the earlier possible starting time. This ideal priority is given by

$$\frac{LLP_j}{LCP},$$

where LLP_j is the longest length path from the beginning of job j to the end of the project and LCP is the length along the critical path of the project.

The factor that adjusts the priority to account for the gene values of the random key chromosome is given by

$$\frac{1 + gene_j}{2}.$$

Thus, the decoding expression of each job j is

$$Priority_j = \frac{LLP_j}{LCP} \times \left[\frac{1 + gene_j}{2}\right]$$

In [65], the minimization of the makespan is the only criterion addressed. In this work, the concept of ideal priority is modified in order to be more appropriate for solving the multi-objective observation scheduling problem. Hence, the ideal priority gives highest priority to select and schedule the acquisition which has the earlier possible starting time. This ideal priority is given by

$$\frac{Tmax_L - Tmin_j}{Tmax_L},$$

where $Tmax_L$ is the latest starting time of the last possible acquisition and $Tmin_j$ is the earliest starting time of acquisition j.

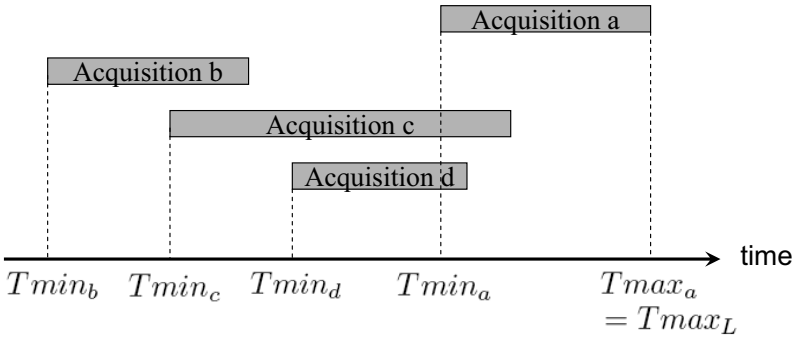
The same factor as in [65] is used to adjust the priority. Thus, the expression of decoding of gene value with ideal priority combination for each acquisition j is

$$Priority_j = \frac{Tmax_L - Tmin_j}{Tmax_L} \times \left[\frac{1 + gene_j}{2}\right].$$

Example of the ideal priority calculation of the second decoding method is illustrated in Figure 3.4. It is applied to the observation scheduling problem for Earth observing satellites, which needs to select and schedule four acquisitions, which are acquisitions a, b, c, and d. For this example, the order of the acquisitions, which will be considered to be assigned in the sequence according to the ideal priority, is b, c, d, and a.

Since the ideal priority, which is used in the equation of this second decoding, is calculated by using the value of problem data. It can make this decoding method faster to reach the optimal solution. However, one should be careful with the entrapment in a local optimum.

The order to consider each acquisition depends on the priority, which is obtained from the above equations. When considering each acquisition, three imperative constraints must be satisfied: the non-repeated selection of the same strip, time windows, and sufficient transition times. Each considered acquisition can be assigned in the sequence, only if the obtained sequence can satisfy the three constraints. When all acquisitions are considered, a temporary sequence is obtained. Then, the stereo constraints are verified on the temporary sequence. If the temporary sequence satisfies the stereo constraints, the



Acquisition	a	b	c	d
Ideal priority	$\frac{Tmax_L - Tmin_a}{Tmax_L}$	$\frac{Tmax_L - Tmin_b}{Tmax_L}$	$\frac{Tmax_L - Tmin_c}{Tmax_L}$	$\frac{Tmax_L - Tmin_d}{Tmax_L}$

Figure 3.4: Example of ideal priority calculation.

final sequence is obtained. Otherwise, the acquisitions that do not satisfy the stereo constraint are removed and it gives way to rescheduling. The flowchart of these decoding steps is depicted in Figure 3.5.

The example of one solution from the smallest instance of the ROADEF 2003 challenge is shown in Figure 3.6. The instance consists of two strips. Thus, the chromosome size is equal to four. This example shows the solution, which is decoded from the basic decoding. The decoding step is used to obtain the sequence of the selected acquisitions and the value of objective function.

The basic decoding and the decoding of gene value with ideal priority combination are experimented on the observation scheduling problem for an agile Earth observing satellite. The obtained results will be illustrated and discussed in Chapter 5.

3 BRKGA applied to multi-user observation scheduling problem for an agile Earth observing satellite

The multi-user observation scheduling problem for an agile Earth observing satellite is considered in this work. Requests coming from several users are considered. Two objective functions have to be optimized and the imperative

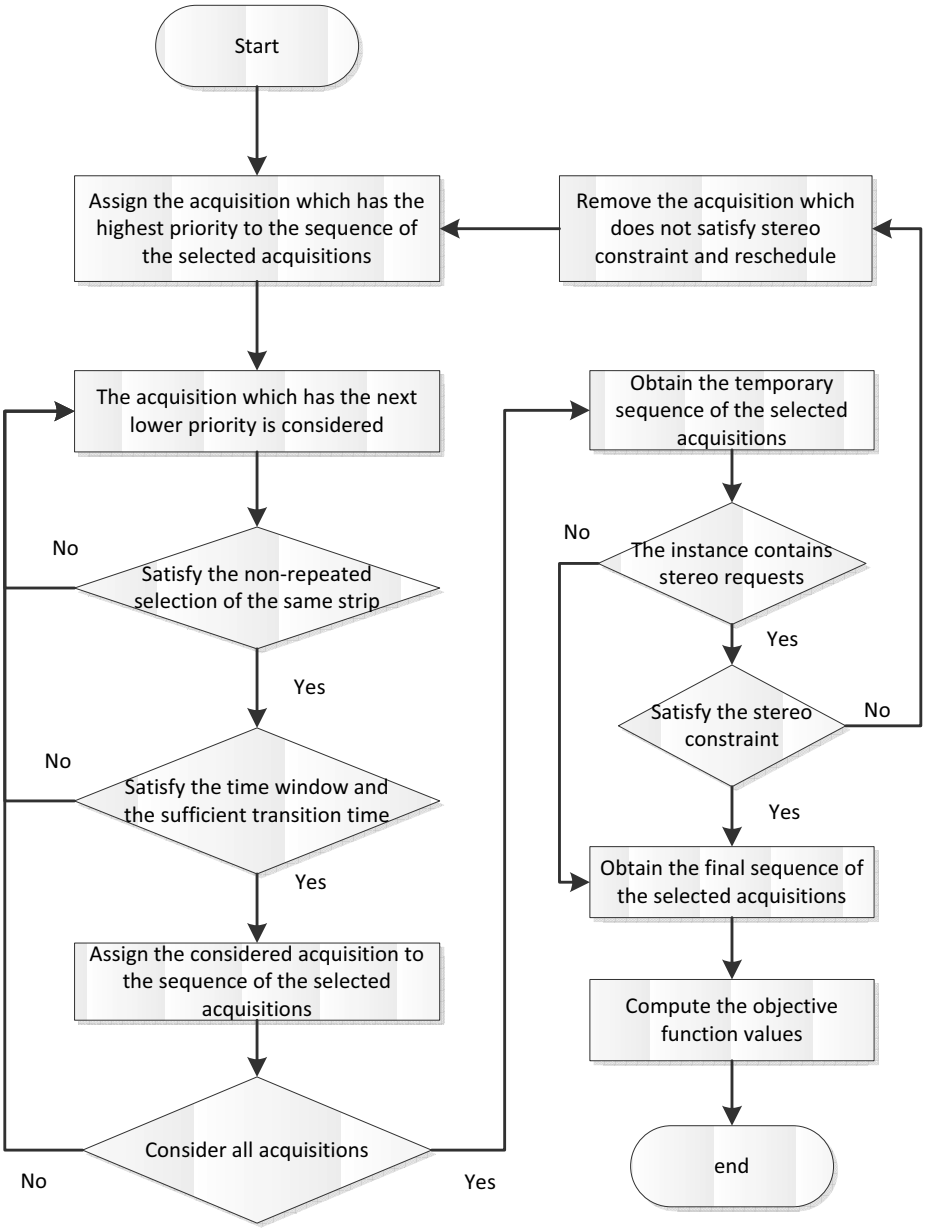


Figure 3.5: Flowchart of constraint verification and acquisition assignment.

constraints must be satisfied as follows:

maximize total profit

minimize maximum profit difference between users

satisfy time windows

sufficient transition times

one of two directions can be acquired for each strip

stereo constraints.

Random-key chromosome	**Acquisition1** | **Acquisition2** | **Acquisition3** | **Acquisition4** 0.6984 | 0.9939 | 0.6885 | 0.2509
Sequence of the selected acquisitions	Acquisition 2 | Acquisition 3
Total profit	**1.04234E+07**

Figure 3.6: Solution example from the instance, which needs to schedule two strips.

After solving this problem, the BRKGA will output the sequences of the selected acquisitions, which are decoded in the decoding step. Then, the obtained profit for each user can be calculated and the objective function values are computed. Both objective functions have to be considered, thus the selection methods for the multi-objective optimization problem must be applied in the elite selection step of BRKGA. They will be presented in Section 3.1. Moreover, this work proposes a hybrid decoding method in Section 3.2. Hybrid decoding combines the use of basic decoding and decoding of gene value with ideal priority. The objective of this hybrid decoding is to bring out the advantages of each decoding. The methods to manage the elite set for the proposed hybrid decoding are studied.

3.1 Selection methods for multi-objective optimization problem

In this chapter, the biased random key genetic algorithm is applied to solve a multi-objective optimization problem. As pointed in Chapter 1, most research in the evolutionary multi-objective optimization area pays attention to develop the selection stage. Because the conflicting objectives, which have to be optimized simultaneously, make the difficulty to judge that a solution is better than the other ones. Therefore, the chromosome selection, which is operated for obtaining the preferred chromosomes in order to become the elite set in BRKGA process, is also not easy. Hence, this work proposes three selection methods to choose p_e preferred chromosomes from the current population to become the elite set in BRKGA process. The three selection methods are borrowed from efficient multi-objective evolutionary algorithms.

3.1.1 Fast nondominated sorting and crowding distance assignment

Fast nondominated sorting and crowding distance assignment methods were proposed in the Nondominated Sorting Genetic Algorithm II (NSGA-II) [25]. In this work, the fast nondominated sorting method is applied to classify the solutions in the population into several ranks. The solutions of Rank 1 or the nondominated solutions are used in this step. The number of nondominated solutions is compared to the parameter setting value of maximum size of elite set. If the number of nondominated solutions is less than or equal to the setting maximum size of elite set, all nondominated solutions will become the elite set. Otherwise, the crowding distance assignment method is applied to select some solutions from the nondominated set. All solutions in the nondominated set are sorted depending on the objective values from high to low. The solutions in the boundary points, which obtain the highest and lowest objective values, are always selected. For each remaining solution i of the considered problem, the estimate density of solutions surrounding is calculated by

$$\begin{aligned} \mathcal{I}\ distance(i) = \quad & [(obj_1(i-1) - obj_1(i+1))/(max_1 - min_1)] \\ & + [(obj_2(i-1) - obj_2(i+1))/(max_2 - min_2)]\,, \end{aligned}$$

where $\mathcal{I}\ distance$ is the crowding distance value, obj_1 refers to the first objective which is the total profit, and obj_2 refers to the second objective, which is the maximum profit difference between users. The process for calculating the crowding distance values for selecting the preferred solutions by using crowding distance assignment is shown in Figure 3.7. When the crowding distance values of all remaining solutions are calculated, the solutions are sorted by the decreasing crowding distance values. The $p_e - 2$ solutions, which have the highest crowding distance values, are selected. These solutions are included to the two pre-selected solutions from the boundary points, and then they are the members of the elite set in BRKGA process.

3.1.2 $\mathscr{S}$ *metric selection* evolutionary multi-objective optimization algorithm

$\mathscr{S}$ *metric selection* evolutionary multi-objective optimization algorithm or SMS-EMOA, which was proposed in [12], is applied to select some solutions in the current population to become the elite set. In this work, we use SMS-EMOA with combining the fast nondominated sorting from NSGA-II. The fast nondominated sorting is applied in order to find the nondominated solutions and SMS-EMOA is used as a selection criterion for limiting the size of elite set. For SMS-EMOA, the solutions in the nondominated set are sorted depending on the objective function values. Then, the hypervolume $\Delta_{\mathscr{S}}(i, \mathscr{R}_1)$ is calculated for each solution i in the nondominated set. It is the hypervolume, which is dominated by solution i, but is not dominated by the

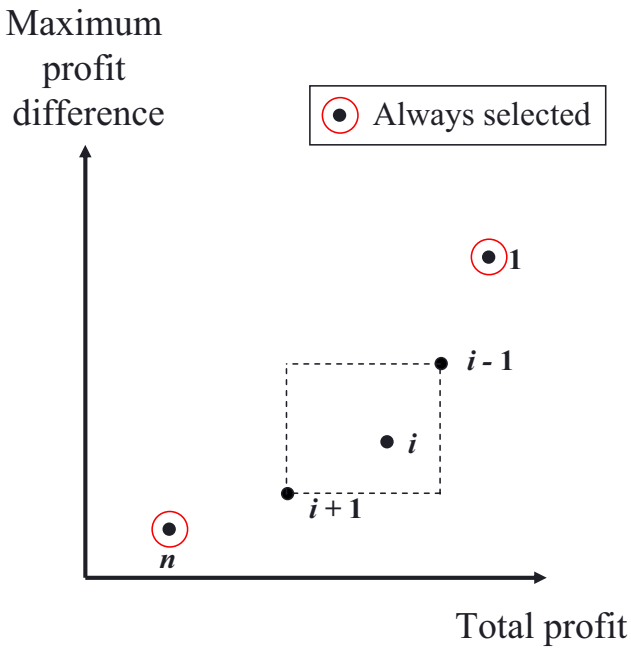


Figure 3.7: The process for calculating the crowding distance values for selecting the preferred solutions by using crowding distance assignment.

other nondominated solutions. The reference point, which is used to calculate the hypervolume, is $(-\infty, \infty)$. The hypervolume of the considered problem is given by

$$\Delta_{\mathscr{S}}(i, \mathscr{R}_1) = (obj_1(i) - obj_1(i+1)) \times (obj_2(i-1) - obj_2(i)),$$

where obj_1 refers to the first objective, which is the total profit, and obj_2 refers to the second objective, which is the maximum profit difference between users. The process for calculating the hypervolume $\Delta_{\mathscr{S}}(i, \mathscr{R}_1)$ is shown in Figure 3.8. The selection discards the solutions that have the least values of hypervolume $\Delta_{\mathscr{S}}(s, \mathscr{R}_1)$, until the number of remaining solutions in the nondominated set is equal to the limit size of elite set and the remaining solutions will become the elite set.

3.1.3 Indicator-based evolutionary algorithm based on the hypervolume concept

The use of an indicator based on the hypervolume concept was proposed in the Indicator-Based Evolutionary Algorithm or IBEA [92]. The indicator based method is used to assign fitness values based on the hypervolume concept to the population members. Then, some solutions in the current population

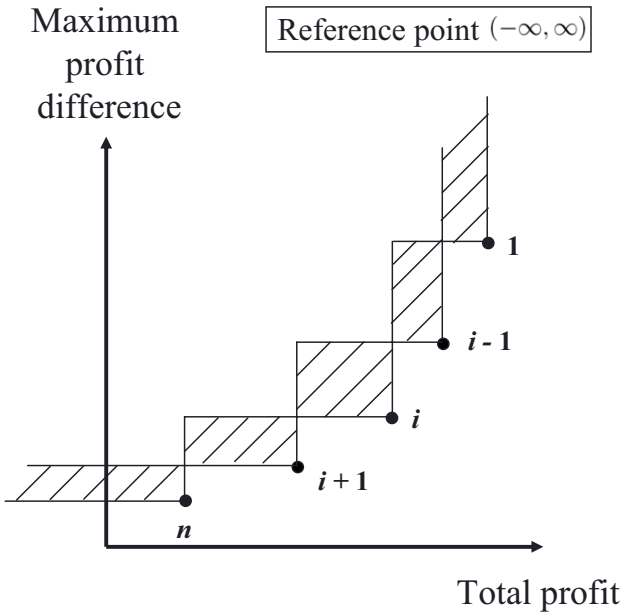


Figure 3.8: Example for calculating the hypervolume $\Delta_{\mathscr{S}}(i, \mathscr{R}_1)$.

are selected to become elite set for the next population. The indicator based hypervolume performs binary tournaments for all solutions in the current population.

The adaptive IBEA is applied in this work, thus the objective values are scaled in the interval $[0, 1]$ by using the upper and lower bound values of each objective. However, the reference point should not be set equal to $(0, 1)$, because the hypervolume value of extreme points on the objective space cannot be calculated. Hence, for the considered problem, which is to maximize the first objective and minimize the second objective, the reference point to calculate the hypervolume of each solution is set equal to $(-1, 2)$ as shown in Figure 3.9.

The indicator value (I_{HD}) can be computed as

$$I_{HD}(\{x^2\}, \{x^1\}) = \begin{cases} I_H(\{x^1\}) - I_H(\{x^2\}) & \text{if } x_1 \prec x_2, \\ I_H(\{x^1\} + \{x^2\}) - I_H(\{x^1\}) & \text{otherwise,} \end{cases}$$

where I_H is the dominated hypervolume of the objective function space. Then, the fitness of all solutions are calculated and its expression is given by

$$F(x^1) = \sum_{x^2 \in P \setminus \{x^1\}} -e^{-I_{HD}(\{x^2\}, \{x^1\})/(c \cdot \kappa)}.$$

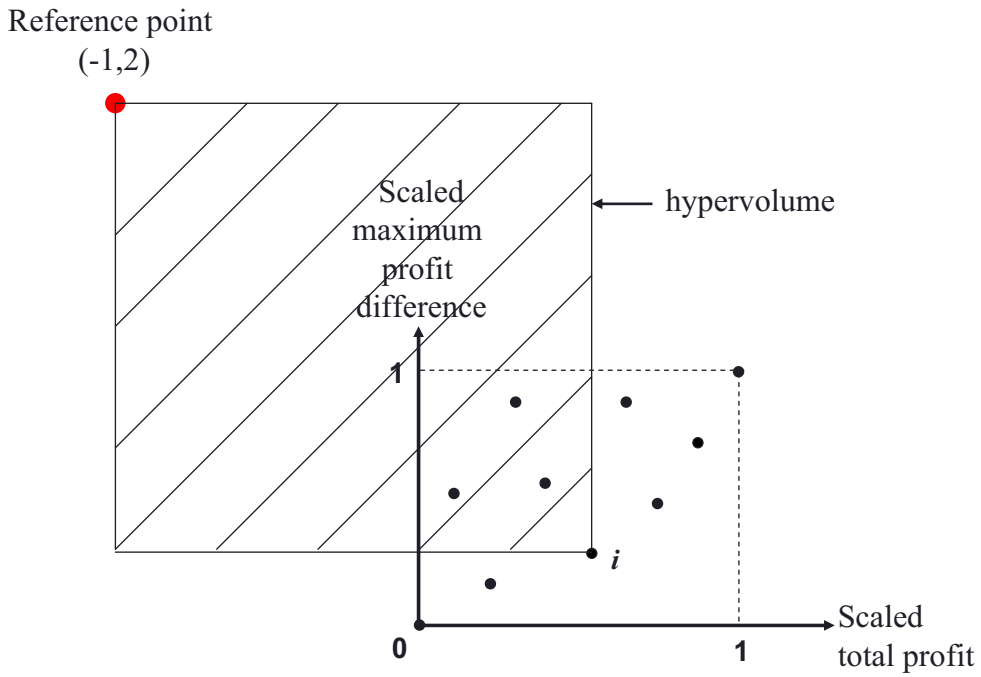


Figure 3.9: The reference point for calculating the hypervolume.

The κ value is set equal to 0.05 regarding the experimental parameter setting in [92]. The c value is calculated as

$$c = \max \left| I_{HD} \left(\{x^1\}, \{x^2\} \right) \right|,$$

where $x^1, x^2 \in P$. The selection is implemented by removing the worst solution with the least fitness value from the population and updating the fitness values of the remaining solutions. The worst solution is removed repeatedly until the number of remaining solutions satisfies the recommended size of elite set for BRKGA.

In this work, the three selection methods are applied to select the solutions in order to become the elite set for the considered problem. The obtained results of these three methods will be compared and discussed in Chapter 5.

3.2 Proposed decoding method: Hybrid decoding

In Section 2.3, the two decoding methods, which are basic decoding and decoding of gene value with ideal priority combination, were presented. They are used in BRKGA process in order to decode a random key chromosome to become one feasible solution. In this decoding step, the efficient decoding

method can give a good solution for the problem and vice versa. Hence, the specification of decoding method is one of the important parts for BRKGA.

In addition to the two presented decoding methods, a hybrid decoding method is proposed in this section. The key difference from the previous decoding methods is that the hybrid decoding method may give more than one feasible solution from the decoding of one chromosome. Because of this, the hybrid decoding can make BRKGA process more efficient for solving multi-objective optimization problems, which need a set of solutions, not only one.

The hybrid decoding combines the uses of the basic decoding, which is given by

$$Priority_j = gene_j$$

and the decoding of gene value with ideal priority combination, which is given by

$$Priority_j = \frac{Tmax_L - Tmin_j}{Tmax_L} \times \left[\frac{1 + gene_j}{2}\right]$$

where $Tmax_L$ is the latest starting time of the last possible acquisition and $Tmin_j$ is the earliest starting time of acquisition j. It obtains two feasible solutions from the decoding of one chromosome.

When applying the hybrid decoding, the methods for management the elite set must be defined. Three methods are used to select the preferred chromosomes to be included into the elite set.

1. Elite set management - Method 1

   Each chromosome in the population is decoded from both decoding methods. The first solution is obtained from the basic decoding and the second solution is obtained from the decoding of gene value with ideal priority combination. Then, both solutions, with corresponding objective function value, are compared using the dominance relation in the Pareto sense. If a solution can dominate the other one, the dominant solution is selected to be stored in the set of solutions. Otherwise, one of the two solutions is selected randomly. The decoding process is repeated until all chromosomes in the population are decoded. When it finishes, the size of solution set is equal to p. Then, the p_e solutions are selected to become the elite set by using the same methods with only one decoding. The principle of elite set management - method 1 is shown in Figure 3.10.

2. Elite set management - Method 2

   All chromosomes in the population are decoded by using the two decoding methods. Two solutions are obtained from the decoding of one chromosome. Both of them are stored in the solution set. Hence, the

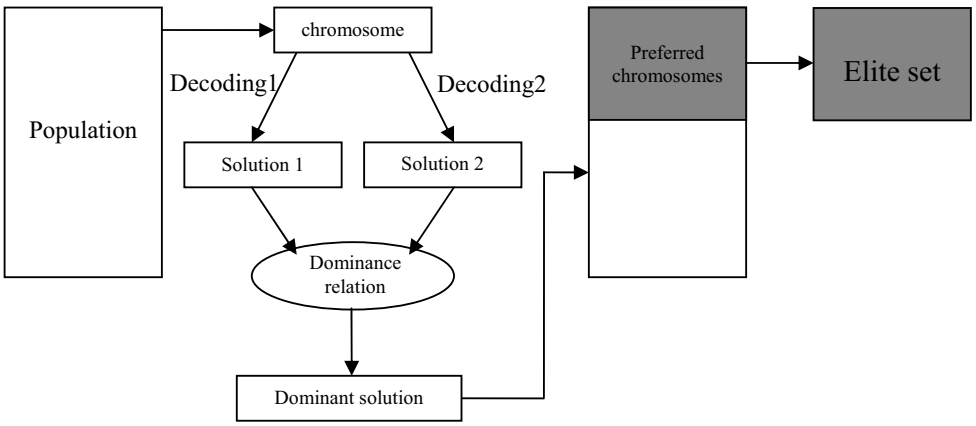


Figure 3.10: Elite set management for hybrid decoding - method 1.

size of solution set is equal to $2p$, when all chromosomes from the current population are decoded. Then, the p_e solutions are selected from the solution set to become the elite set. The principle of elite set management - method 2 is shown in Figure 3.11.

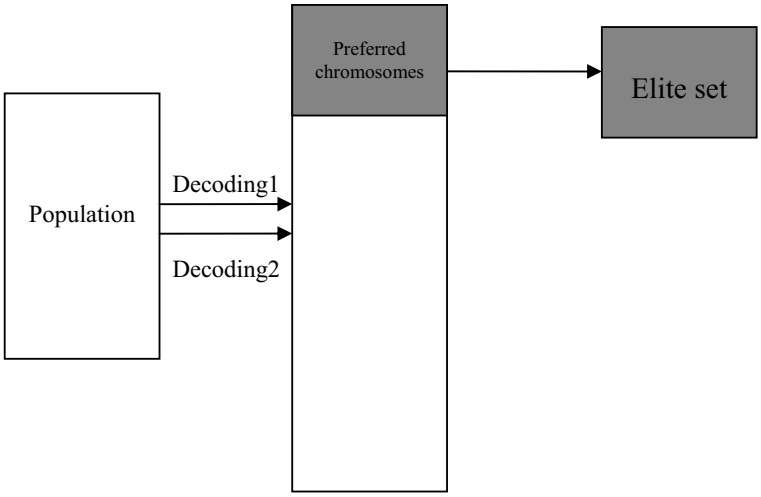


Figure 3.11: Elite set management for hybrid decoding - method 2.

3. Elite set management - Method 3

   Each chromosome in the population is firstly decoded by using the priority equation of basic decoding and the obtained solution is stored in the first solution set. Similarly, the same chromosome is decoded by

using the priority equation of the decoding of gene value with ideal priority combination. Then, the obtained solution from this decoding is stored in the second solution set. When all chromosomes in the population are decoded and the solutions are stored into two solution sets, the selection methods are applied to select p_e solutions for becoming the elite set. Hence, the $p_e/2$ preferred solutions must be chosen from each solution set as shown in Figure 3.12.

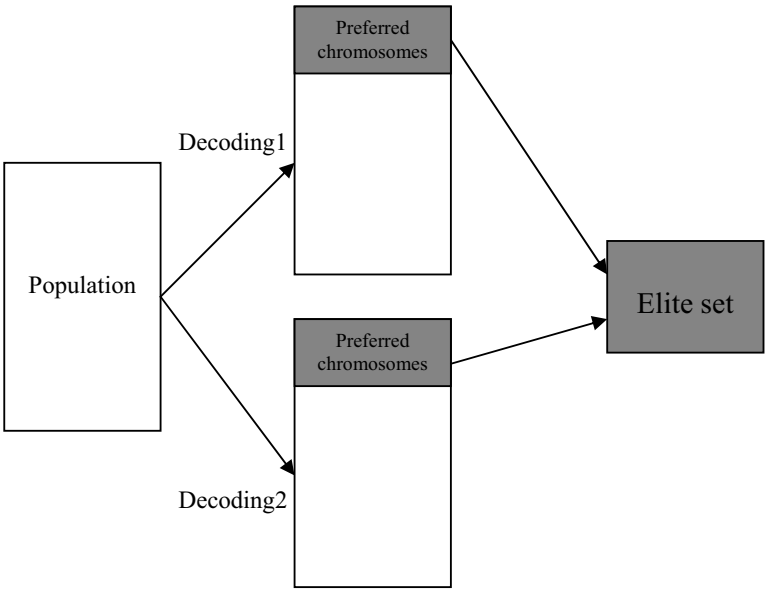


Figure 3.12: Elite set management for hybrid decoding.

The three elite set management methods for the proposed hybrid decoding are evaluated on modified instances issued from ROADEF 2003 challenge. The obtained results from the hybrid decoding will be reported in Chapter 5. They are also compared with the results, when using only one decoding method.

4 Conclusion

In this chapter, the biased random key genetic algorithm or BRKGA is applied to solve the multi-user observation scheduling problem for an agile observing satellite. The details of BRKGA process are explained. In the encoding step, random key chromosomes are generated. They can be operated by the genetic algorithm operators. Each chromosome contains several genes, which represent an acquisition. The genetic algorithm process is described; it is used to

generate the new population for each iteration. The new population is generated from three sets: elite set, crossover offspring set, and mutant set. To obtain a solution, the chromosome has to be decoded. The solution is the sequence of the selected acquisitions. Two decoding methods are presented. However, the problem in this work is modeled by considering two objectives. Hence, the selection methods for multi-objective optimization are applied to use in the elite selection step. These methods were proposed in the existing popular multi-objective evolutionary algorithms. Furthermore, a hybrid decoding is also proposed in this work. Two decoding methods are combined in order to increase the performance of BRKGA. Experiments are conducted on realistic instances. The results will be reported and discussed in Chapter 5.

Indicator-based multi-objective local search

4

1 Introduction

This chapter proposes the use of indicator-based multi-objective local search (IBMOLS) to solve the multi-user observation scheduling problem for an agile observing satellite. IBMOLS was proposed in [7]. It is a generic algorithm, which combines the use of basic local search and a binary indicator of indicator-based evolutionary algorithm.

This chapter is organized as follows. Section 2 presents IBMOLS and its adaption to the multi-user observation scheduling problem for an agile Earth observing satellite. The neighborhood structure is proposed in Section 3. The procedure for feasibility checking is described in Section 4.

2 IBMOLS applied to multi-user observation scheduling problem for an agile Earth observing satellite

In single objective optimization, a local search is a metaheuristic method, which starts from an initial solution and move to a neighbor improving this solution. The process is iterated until a local optimum is found. In many real world applications, the local search was applied to solve the problems and good solutions have been obtained, even for large-size problems.

In this work, IBMOLS is applied to solve the problem, which is bi-objective. IBMOLS is a population-based iterated local search method. The local search is used to explore the neighborhood for finding the solutions, which are better than the current ones. Moreover, IBMOLS uses a binary

indicator as the selection operator. Indeed the binary indicator has a main advantage, which is no requirement of additional diversity preservation mechanism.

The standard of IBMOLS was presented in Chapter 1. However, some steps are modified according to the considered problem. The steps of IBMOLS, which is applied for this work, are explained. In Section 2.1, the overview of IBMOLS procedure is described. Then, the initial generation for the first iteration and for the other iterations of the iterated local search are presented in Sections 2.2 and 2.3. Finally, Sections 2.4 and 2.5 present the fitness computation and the local search step, respectively. The local search step describes the neighborhood exploration and how to select the neighbor.

2.1 Overview of IBMOLS

IBMOLS is an iterated local search. For the first iteration, the approximate Pareto front PO is generated as an empty set and it is updated, whenever each iteration is finished. In each iteration of IBMOLS, it starts by generating the initial population. Two types of initial population generation are presented in this work. The first one is applied for the first iteration and the second one is applied for the other iterations. The details of the first and the second types will be explained in Sections 2.2 and 2.3, respectively. Then, the nondominated solutions in the population are stored in the archive set A. The fitness values of all individuals in the population are computed and the local search step is applied for each individual. The details of fitness computation and the local search step will be described in Section 2.4 and Section 2.5, respectively. After that, the updated population is combined with the archive set A and the nondominated solutions of this combined set are stored in the new archive set A. If the archive set A changes, the process returns to apply the local search step. Otherwise, this iteration is finished and the archive set A is obtained. Then, the approximate Pareto front PO will be updated by combining the obtained archive set A with the approximate Pareto front PO and the set of nondominated solutions from the combined set becomes the new approximate Pareto front PO. The next iteration is continued, if it does not satisfy the stopping criteria. The flowchart of the overview IBMOLS is illustrated in Figure 4.1.

2.2 Initial population generation - first iteration

The IBMOLS is a population-based iterated local search that needs N individuals to be generated to become the initial population. Each individual represents one solution, which is a sequence of the selected acquisitions including the starting time for acquiring each acquisition. This work uses two

methods for generating the initial population for the first iteration. The first method generates the initial population randomly. The second one also uses data of the problem instances for generating some initial individuals.

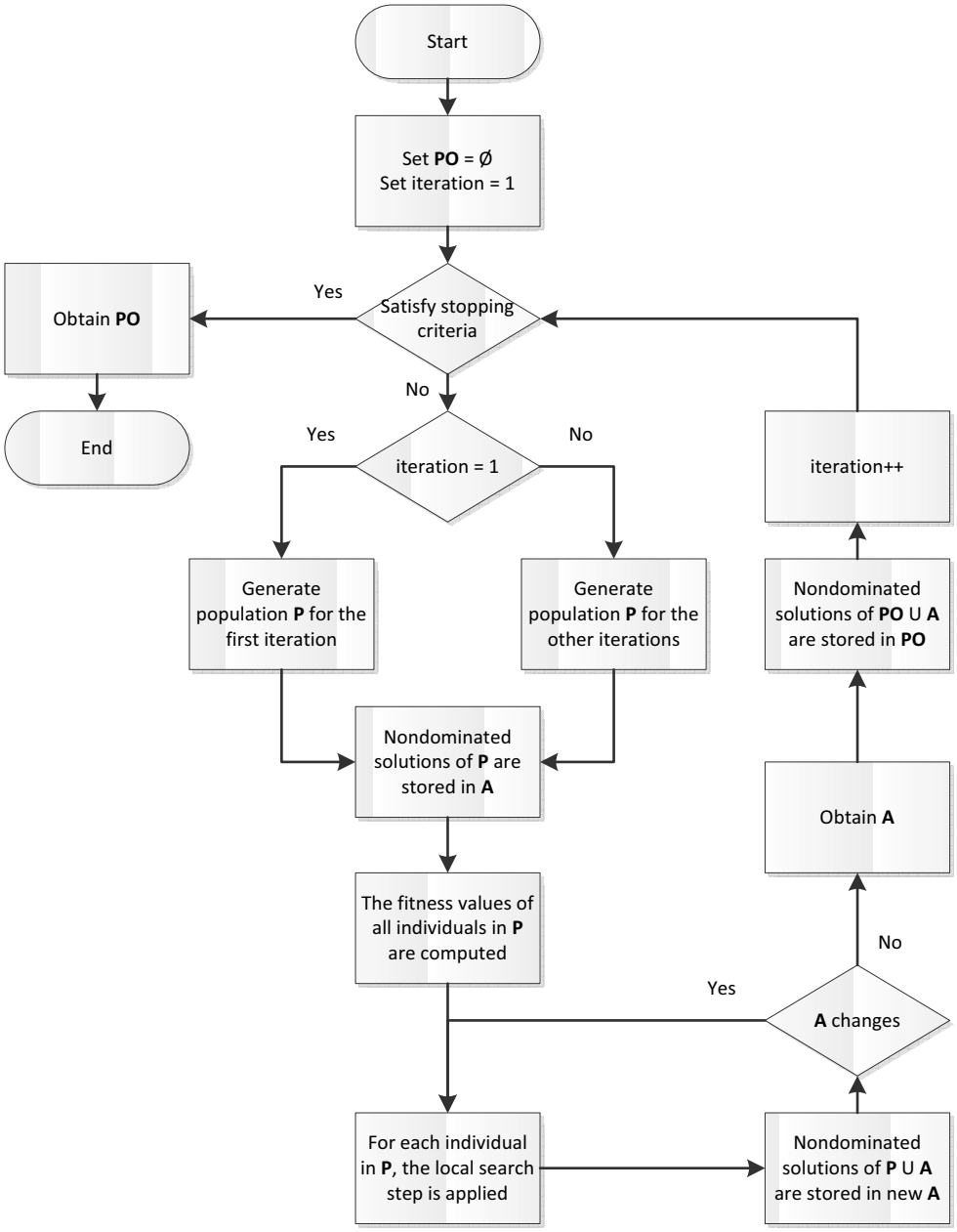


Figure 4.1: Flowchart of the overview IBMOLS.

2.2.1 Random generation

For the random generation, the flowchart of the initial population generation is shown in Figure 4.2. All acquisitions are assigned to be the members of the selected acquisition set depending on a random order. For each acquisition, it is checked that it satisfies the sufficient transition time constraint and the time window constraint. If it satisfies both constraints, the satisfied starting time is set in the starting time set. Moreover, the acquisition, which comes from the same strip of the considered acquisition, is removed from the selected acquisition set. Otherwise, the considered acquisition is removed. The process for checking these constraints is repeated until all acquisitions in the selected acquisition set are tested. After that, the temporary selected acquisition set and starting time set are obtained. In this step, the stereo constraint has to be checked for each selected acquisition one by one in the temporary set. If the checking acquisition comes from the stereo strip, its twin must also be assigned. If its twin is not assigned, the checking acquisition is removed. When all selected acquisitions are checked, the starting time set is re-computed.

2.2.2 Using useful data of problem instances

This section presents the second method for generating the initial population of the first iteration. As presented previously, in this work, IBMOLS process needs to generate N individuals for becoming the initial population. This method generates five individuals by using useful data of problem instances and other five individuals are generated by random generation, as presented in Section 2.2.1. This section describes the five strategies. Each strategy depends on an order which is listed below:

1. The earliest starting time ($Tmin$)
2. The time interval between the earliest starting time and the latest starting time ($Tmax - Tmin$)
3. The obtained profit ($Profit$)
4. The fraction of the obtained profit and the earliest starting time ($Profit/Tmin$)
5. The fraction of the obtained profit and the time interval between earliest starting time and the latest starting time ($Profit/\left(Tmax - Tmin\right)$)

The orders for considering each acquisition in order to generate the five individuals are set as in Figure 4.3. For the first order, the acquisition, which has the smallest earliest starting time, should be scheduled in the beginning

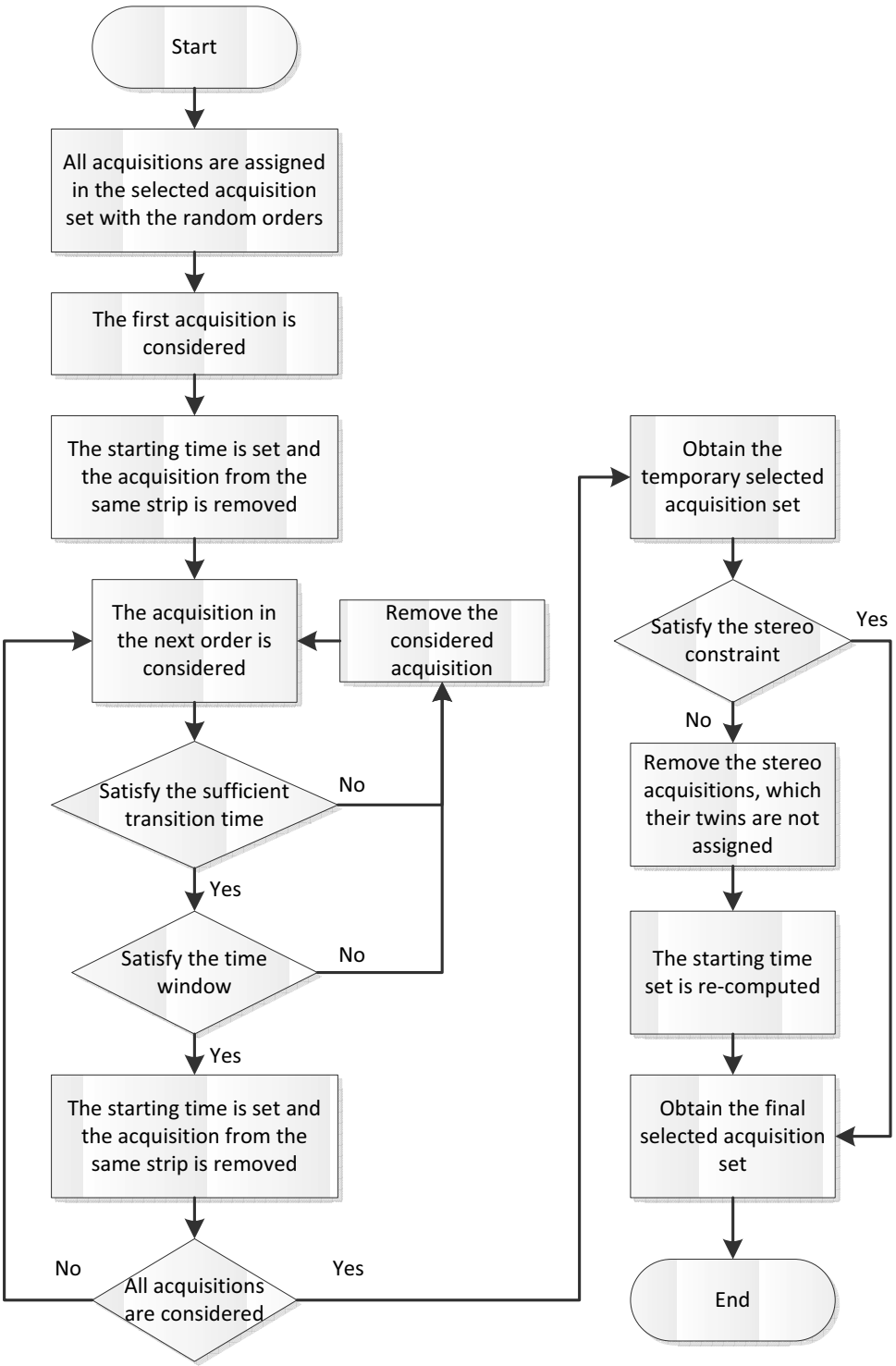


Figure 4.2: Flowchart of the initial randomly generated population.

of the sequence. The second order gives the chance for assigning the acquisition which has the smallest possible starting time interval, because the acquisition with the larger possible interval can be more easily assigned. For the third order, the profit of each acquisition is computed, in the case that the acquisition is selected and scheduled in the sequence and the whole area of its associated request is acquired. For this order, the acquisition, which gives the highest profit, should be selected firstly. The fourth and fifth orders are extended from the three first ones. By the fourth order, it combines the first and the third orders. The fifth order combines the second and the third ones.

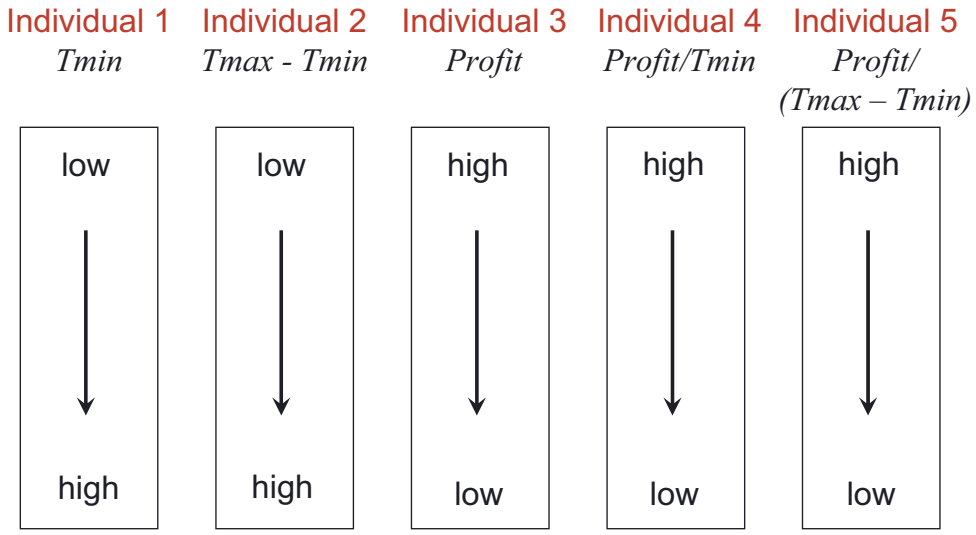


Figure 4.3: The orders assignment for considering each acquisition in order to generate the five individuals.

Each individual consists of a selected acquisition set and a starting time set, that each element of the starting time set illustrates the starting time for acquiring the according selected acquisition. Each individual is generated from one of the five presented strategies, which are used to define the considering orders of each acquisition. The verification process is the same as the process of the initial randomly generated population, but the order for considering each acquisition is specified by one of the five strategies, not randomly generated. Thus, the flowchart of the initial population generation by using the useful data of problem instances is shown in Figure 4.4.

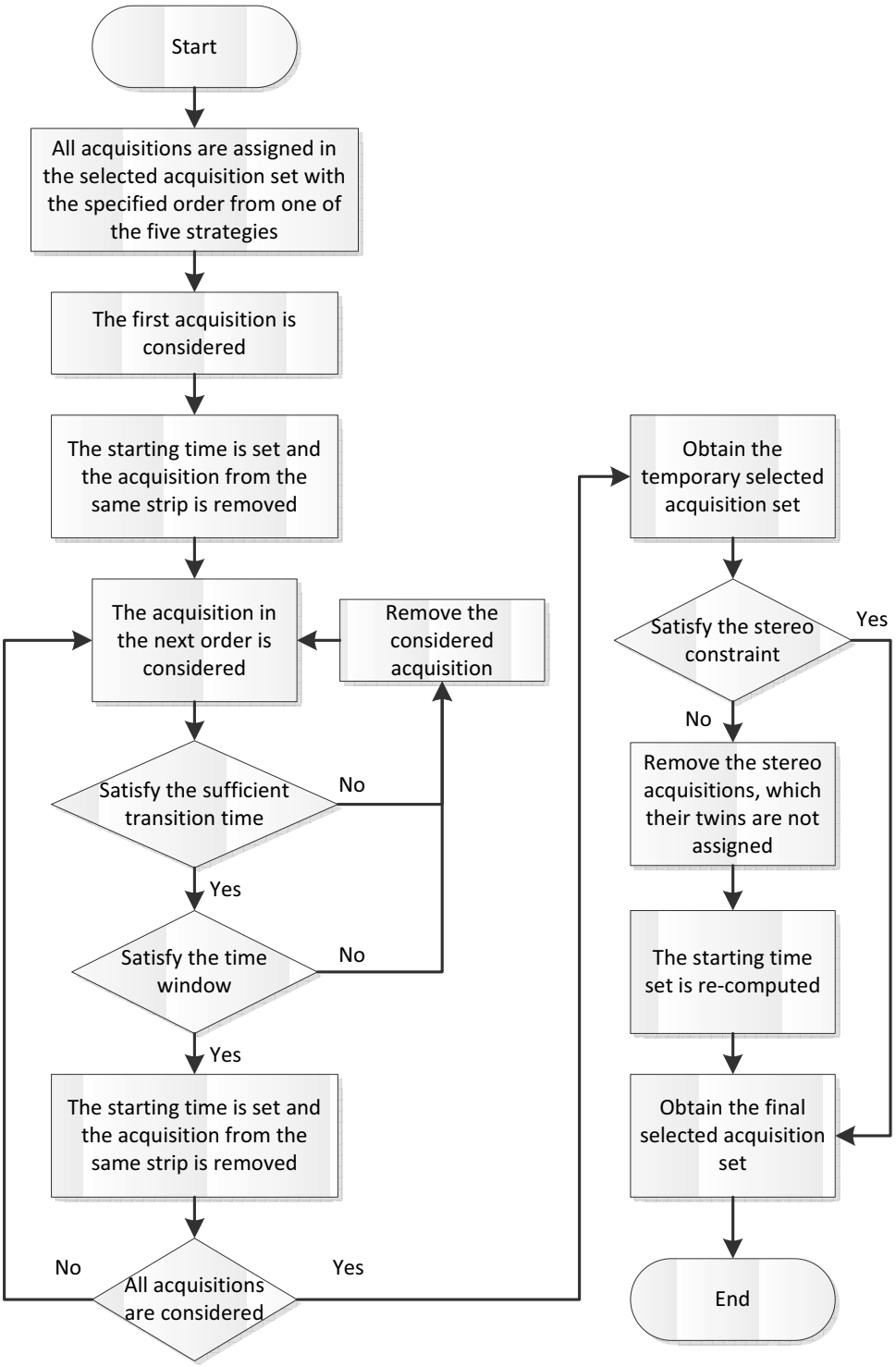


Figure 4.4: Flowchart of the initial population generating by using the useful data of problem instances.

2.3 Initial population generation - other iterations

In IBMOLS, an iterated local search is used for searching for the nondominated solutions by starting the search from different initial populations. The initial population generation of the first iteration has been described in Section 2.2. Now, the initial population generation for the other iterations is explained in this section. Two methods are applied in this work.

2.3.1 Random generation

The random generation in order to build the initial population for the other iterations, starting from the second iteration, uses the same method than for the first iteration. For more detail, see in Section 2.2.1.

2.3.2 Perturbation

This method uses a perturbation mechanism to generate the initial population starting from the second iteration. For the perturbation, an individual is generated by modifying a solution from the approximate Pareto front PO of the current iteration. In this work, the solutions in the approximate Pareto front are randomly selected. The number of selected solutions is equal to the size N of initial population. Each solution consists of the selected acquisition set ori_acq_sel and the starting time set ori_start_time. They are modified by removing some acquisitions in the random position j from the selected acquisition set and also removing the starting time elements of the associated removed acquisition. The number of the removing elements is about $1/4$ of the size n_{ori} of the original selected acquisition set. Moreover, during removing, the stereo constraint has to be verified. If the removed acquisition is a part of the stereo request, its twin must also be removed. The acquisition removing is repeated until the number of the remaining aquisition in the selected acquisition set or n_{modi} is less than or equal to $3/4$ of the size n_{ori} of the original set. Then, the modified selected acquisition set $modi\_acq\_set$ and the modified starting time set $modi\_start\_time$ will become the parts of the individual, which is a member of the initial population in the next iteration. The initial population generation by using the perturbation is shown in Algorithm 7. The example of a modified individual, which becomes a member of the initial population by using the perturbation is illustrated in Figure 4.5. In this example, the size of the original selected acquisition set and the starting time set from the approximate Pareto front is equal to eight. Hence, two random acquisitions are removed from the original one.

For the avoidance to generate any solution, which has been already visited, we must be careful for generating the individual to become the initial population. In the process of perturbation, the number of acquisitions, which

Algorithm 7 Procedure of the initial population generating by using the perturbation

for $i = 1$ **to** $i = N$ **do**

Step 1: Original individual selection
Select randomly an individual from the approximate Pareto front PO.
The selected individual consists of ori_acq_sel and ori_start_time.

Step 2: Element removing
repeat

Step 2.1: Select randomly the removing position j.
Step 2.2: Remove the acquisition in position j.
Step 2.3: Remove the starting time in position j.
Step 2.4: Verify the stereo constraint.
if The removed acquisition is a part of stereo request. **then**

Remove the twin of removed acquisition.
Remove the starting time of the twin.
end if
until $n_{modi} \leq \frac{3}{4} n_{ori}$

Step 3: Counter increment
$i \leftarrow i + 1$
end for

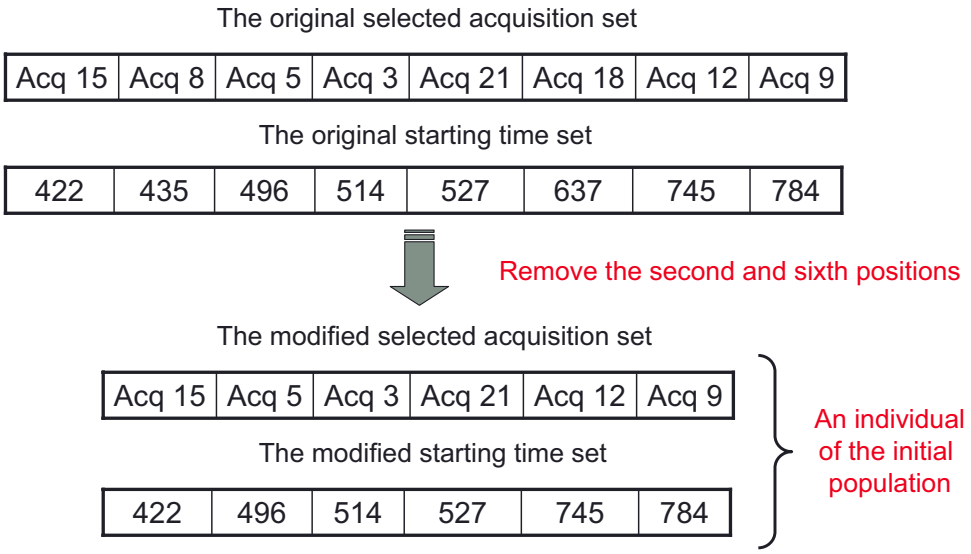


Figure 4.5: The example of a modified individual, which becomes a member of the initial population by using the perturbation.

have to be removed in order to generate the individual, is pre-computed. Some elements of the original selected acquisition set and the original starting time set will be removed, only if, the pre-computed number of the removed acquisitions is more than or equal to two. Otherwise, the perturbation will generate the individual, which has been visited. In the case that the number of removed acquisitions is equal to zero, the same individual from the approximate Pareto front will be generated. If the number of removed acquisitions is equal to one, this individual can be generated by exploring the neighborhood. The neighborhood structure will be explained later in Section 3. Hence, the perturbation will not be applied, if the number of removed acquisition is less than two. The random generation will be used to generate the individual, instead of using the perturbation.

2.4 Fitness computation

After the initial population generation, the fitness value of each individual has to be computed. In this work, the indicator-based evolutionary algorithm based on the hypervolume concept from [92] is used to compute the fitness. This method was also applied to select the preferred individuals in BRKGA, which was presented in Chapter 3. The indicator performs binary tournaments for all individuals in the population P.

The objective values of each individual are computed and they are scaled in the interval $[0, 1]$ by using the upper bound and lower bound of each objective. As in [92], the upper bound ($\overline{b}$) is

$$\overline{b_i} = \max_{x \in P} f_i(x)$$

and the lower bound ($\underline{b}$) is

$$\underline{b_i} = \min_{x \in P} f_i(x),$$

where f_i is an objective function i. The worst values of the two objectives, which are 0 and 1 can be chosen to be the reference point for computing the hypervolume. However in this work, we set the reference point equal to $(-1, 2)$, since the hypervolume of the two extreme points can be computed. Then, the indicator value based hypervolume (I_{HD}) can be computed as

$$I_{HD}(\{x^2\}, \{x^1\}) = \begin{cases} I_H(\{x^1\}) - I_H(\{x^2\}) & \text{if } x_1 \prec x_2, \\ I_H(\{x^1\} + \{x^2\}) - I_H(\{x^1\}) & \text{otherwise,} \end{cases}$$

where I_H is the dominated hypervolume of the objective function space, and the fitness value (F) is

$$F(x^1) = \sum_{x^2 \in P \setminus \{x^1\}} -e^{-I_{HD}(\{x^2\}, \{x^1\})/(c \cdot \kappa)},$$

where κ value is set equal to 0.05 as the experimental parameter setting in [92] and c value is computed as

$$c = \max \left| I_{HD} \left(\{x^1\}, \{x^2\} \right) \right|,$$

where $x^1, x^2 \in P$.

2.5 Local search step

This local search step starts from an individual in the population P and move iteratively to a neighbor by exploring the neighborhood. In this work, the process of neighbor selection for replacing the worst solution is modified from the original IBMOLS. The original IBMOLS uses the first improvement strategy to select the neighbor for replacing the worst solution in the population. However, the best improvement strategy is used in this work. The fitness values of each neighbor, visited by the neighborhood exploration, are computed. Then, the neighbor, which obtains the best fitness, is generated and selected for replacing the worst solution in the population. The local search step works as shown in Algorithm 8.

Algorithm 8 Procedure of the local search step

for All individuals $x \in P$ **do**

Step 1: Neighborhood exploration
Generate all neighbors in the neighborhood $Y = (y_1, y_2, y_3, \ldots, y_n)$.

Step 2: Fitness computation
for All neighbors $y \in Y$ **do**
Compute the fitness value of y_i by using the indicator based hypervolume
end for

Step 3: Best neighbor selection
Select the best neighbor y^* from Y (the neighbor with the highest fitness value).

Step 4: Best neighbor addition
Add the best neighbor y^* to P.

Step 5: Fitness update after neighbor addition
for All individuals $x \in P$ (except y^*) **do**
Update the fitness value $Fit(x) = Fit(x) - e^{-I_{HD}(\{y^*\},\{x\})/(c \cdot \kappa)}$.
end for

Step 6: Worst individual selection
Select the worst individual w from P (the individual with the lowest fitness value).

Step 7: Worst individual removing
Remove the worst individual w from P.

Step 8: Fitness update after individual removing
for All individuals $x \in P$ **do**
Update the fitness value $Fit(x) = Fit(x) + e^{-I_{HD}(\{w\},\{x\})/(c \cdot \kappa)}$.
end for
end for

In the local search step, the neighborhood of all individuals in the population P are explored and the population is updated. The archive set A is updated by storing the nondominated solutions of the combined set $P \cup A$, which combines the updated population P and the previous iteration archive set A. If the updated archive set A does not change, the local search step will be stopped. Otherwise, another local search step is performed.

3 Neighborhood structure

In this section, neighborhood structures are presented. According to the local search step, the best improvement strategy is used in this work. The fitness values of each neighbor in the neighborhood is computed and compared with the best found neighbor. If the fitness of the considered neighbor is better than the fitness of the best found neighbor, the feasibility to generate the considered neighbor is checked. Otherwise, the considered neighbor is discarded. If the considered neighbor is feasible, it is generated and replaces the best found neighbor. The process is repeated until all neighbors in the neighborhood are explored and then the best neighbor is obtained. The procedure for searching for the best neighbor follows the steps as in Algorithm 9. Then, the best neighbor is added in the population and the worst individual in the population is removed. The neighborhood, which is used in this work, consists of six types of move:

1. Insert a mono acquisition i
2. Remove a mono acquisition i
3. Insert twin acquisitions i and j
4. Remove twin acquisitions i and j
5. Replace a mono acquisition i with a mono acquisition j
6. Replace twin acquisitions i and j with twin acquisitions k and l

For each type of move, all mono acquisitions or all pairs of twin acquisitions are tried to be inserted, removed, or replaced one by one. For example, in the insertion of a mono acquisition, all mono acquisitions, which have not been assigned in the original sequence, are tried to be inserted one by one in the sequence. Similarly for the removing of a mono acquisition, each assigned mono acquisition is tried to be removed from the sequence. In this work, the experiments test two groups of types of move for comparing the quality of the obtained results and the computation time. The first group applies all six types of move. The second group applies only the first four types of move,

excluding the replacement of a mono acquisition and twin acquisitions. The details of experiments and the results comparison will be presented in Chapter 5.

Algorithm 9 Procedure of the best neighbor search

Step 1: Initialization
Set the initial best fitness value $Fit(y^*) = -\infty$.

bf Step 2: Neighborhood exploration
for All neighbors y_i in neighborhood Y **do**

Compute the objective function values and the fitness value $Fit(y_i)$.
if $Fit(y_i) > Fit(y^*)$ **then**
Check the feasibility of the neighbor y_i.
if Neighbor y_i is feasible **then**

Generate the neighbor y_i.
$y^* \leftarrow y_i$
$Fit(y^*) \leftarrow Fit(y_i)$
else

Discard neighbor y_i.
Counter increment $i \leftarrow i + 1$
end if
else

Discard neighbor y_i.
Counter increment $i \leftarrow i + 1$
end if
end for

Step 3: Best neighbor obtained
return y^*

4 Feasibility checking

Among the six types of move of the neighborhood structure, when occurring a move that inserts or replaces the acquisition, the feasibility of the neighbor has to be checked. An unassigned acquisition can be inserted in a position of the selected acquisition set, only if, all imperative constraints are satisfied.

For the mono acquisition, the three constraints, which consist of the time window constraint, the sufficient transition time constraint and the constraint that the same strip can be selected and acquired only one direction, must be verified. For the stereo acquisition, the three constraints of the mono acquisition including the stereo constraint have to be checked. In this part, two methods for checking the feasibility of the insertion of a mono unassigned acquisition are explained. The first method checks the feasibility by comparing directly between the possible starting time of the acquisition, which needs to be inserted, and the starting times of the associated acquisitions in the starting time set. For the second method, the latest starting times of the selected acquisitions, which are scheduled behind the insertion position, have to be computed before checking the feasibility. Hence, the variables, which are used to check the feasibility, are:

- the individual x consisting of:
  - the selected acquisition set $acq\_sel = \{sa_1, sa_2, \ldots, sa_{id}, \ldots, sa_N\}$,
  - the starting time set $start\_time = \{ta_1, ta_2, \ldots, ta_{id}, \ldots, ta_N\}$,

  where ta_{id} is the starting time to acquire acquisition sa_{id},
- $Tmin(j)$, which is the earliest starting time, computed from the visibility times of acquisition j ($T_e[j,0], T_l[j,0], T_e[j,1], T_l[j,1]$),
- $Tmax(j)$, which is the latest starting time computed from the visibility times of acquisition j ($T_e[j,0], T_l[j,0], T_e[j,1], T_l[j,1]$),
- $Du(j)$, which is the duration time in order to take acquisition j,
- $Dt(j,l)$, which is the transition time of the camera from the ending position of acquisition j to the starting position of acquisition l.

4.1 Feasibility checking: Method 1

For this method, an unassigned acquisition $Acq\ k$ is considered to be inserted in a position of the selected acquisition set. The possible starting time of acquisition $Acq\ k$ is compared directly with the values in the starting time set. For example, the unassigned acquisition $Acq\ k$ is tried to be inserted in front of the selected acquisition sa_3 as shown in Figure 4.6. If the starting time is possible and there is enough free space for $Acq\ k$ to be inserted, this neighbor is generated. Otherwise, this neighbor will not be generated. The first method of the insertion feasibility checking can be verified in three cases depending on the insertion position, which are:

- the acquisition $Acq\ k$ can be inserted in the first position (position 1) of the selected acquisition set, only if

$$Tmin(Acq\ k) + Du(Acq\ k) + Dt(Acq\ k, sa_1) \leq ta_1$$

- the acquisition $Acq\ k$ can be inserted in the middle position id (position 2 to N) of the selected acquisition set, only if

$$ta_{id-1} + Du(sa_{id-1} + Dt(sa_{id-1}, Acq\ k)) \leq Tmax(Acq\ k)$$

and

$$Tk + Du(Acq\ k) + Dt(Acq\ k, sa_{id}) \leq ta_{id}$$

where $Tk = max(Tmin(Acq\ k), ta_{id-1} + Du(sa_{id-1}) + Dt(sa_{id-1}, Acq\ k))$

- the acquisition $Acq\ k$ can be inserted in the last position (position $N+1$) of the selected acquisition set, only if

$$ta_N + Du(sa_N) + Dt(sa_N, Acq\ k) \leq Tmax(Acq\ k)$$

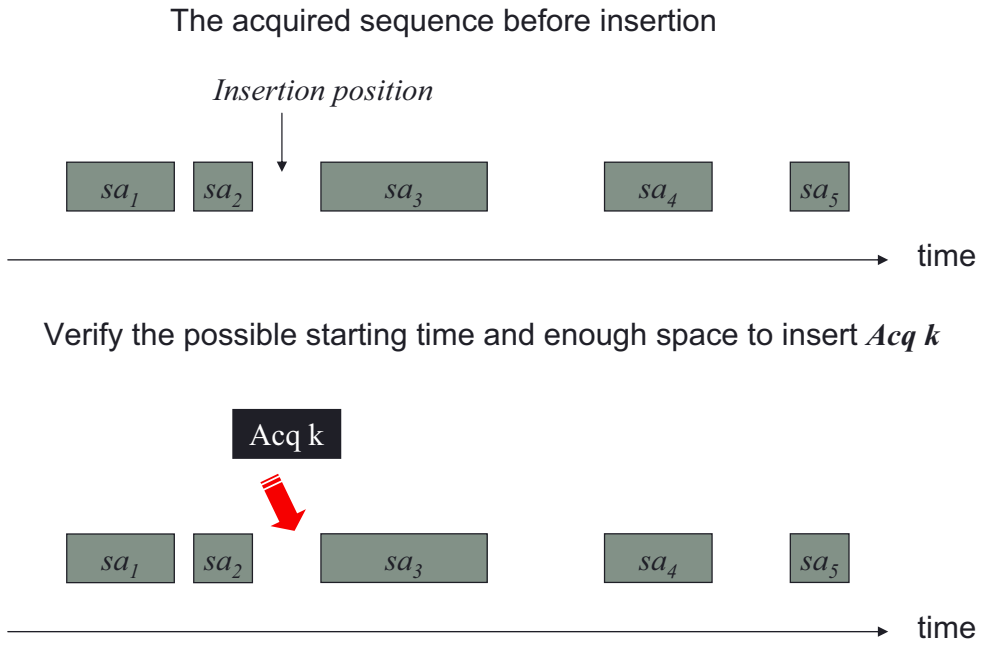


Figure 4.6: Verify the possibility to start for acquiring the acquisition $Acq\ k$ and the enough free space to insert it at the insertion position.

If it is possible to insert $Acq\ k$ in the selected acquisition set, the starting time values in the starting time set are re-computed. Moreover, if the acquisition $Acq\ k$ is a part of the stereo request, the twin of the acquisition $Acq\ k$ must also be checked the insertion feasibility.

4.2 Feasibility checking: Method 2

This section presents the improved method, which can increase the insertion possibility. This second method proposes to compute the latest starting time of the assigned acquisitions, which are scheduled behind the insertion position, before checking the feasibility. This step is included in the process, thus the insertion feasibility checking of this second method has steps as follows:

- Step 1: Compute the latest starting times $Q(id)$

  The free space as large as possible is prepared to support the insertion of the mono unassigned acquisition $Acq\ k$. All positions in the selected acquisition set are considered. The latest starting times $Q(id)$ of each selected acquisition sa_{id} can be computed before checking the possibility. The method is to postpone the acquisitions which are scheduled behind the insertion position. They are scheduled to be taken as late as possible at their latest possible starting times. For example, the unassigned acquisition $Acq\ k$ is tried to be inserted in front of the selected acquisition sa_3. The other acquisitions following sa_3 can be moved to the right as late as possible. This example is shown in Figure 4.7.

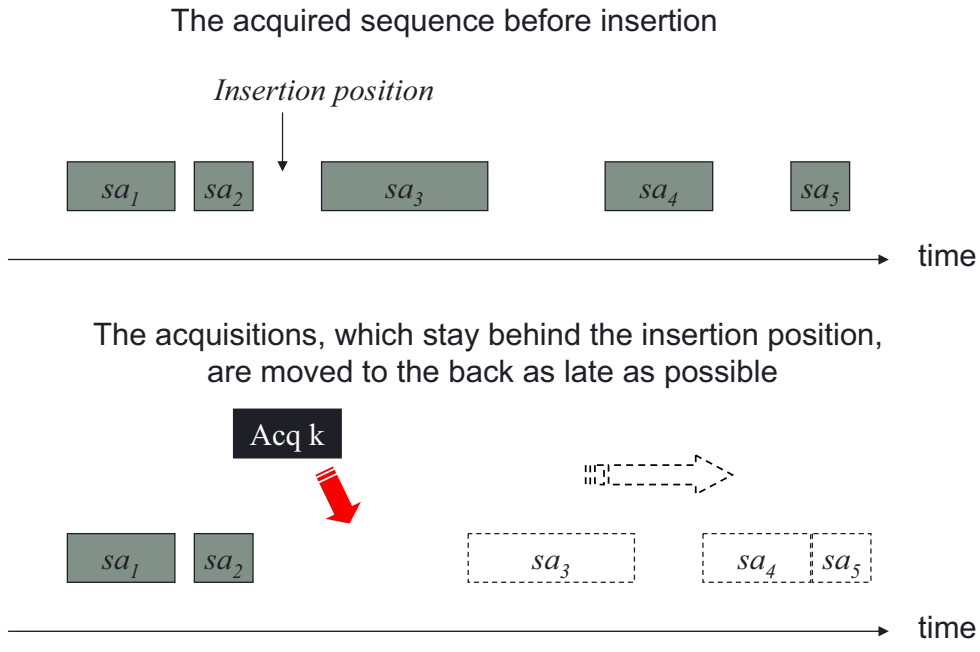


Figure 4.7: The large space is prepared for the insertion of the unassigned acquisition.

For the last acquisition sa_N, its latest starting time $Q(N)$ can be computed by

$$Q(N) = Tmax(sa_N).$$

For the other acquisitions sa_{id}, where $1 \leq id \leq N-1$, their latest starting times $Q(id)$ are given by

$$Q(id) = min(Tmax(sa_{id}), Q(id+1) - Du(sa_{id+1}) - Dt(sa_{id}, sa_{id+1})).$$

Example of the latest starting time $Q(id)$ computation is shown in Figure 4.8.

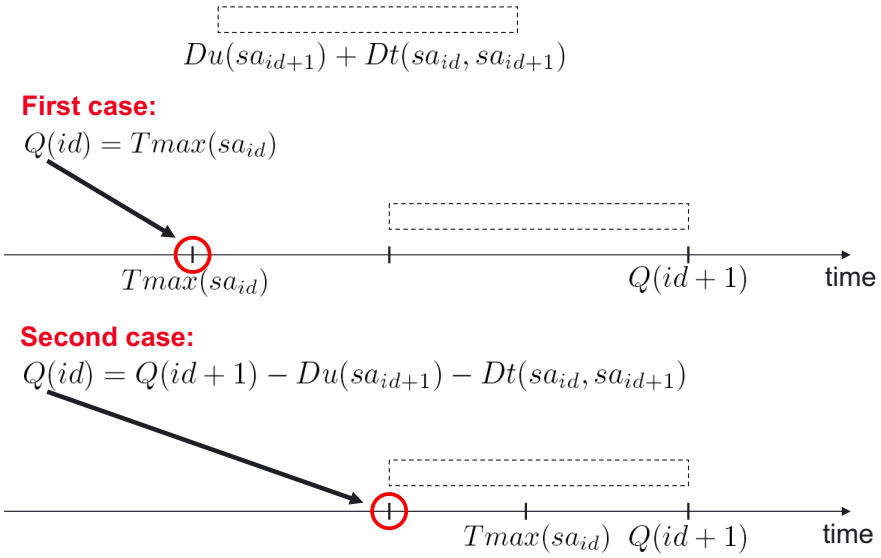


Figure 4.8: Example of the latest starting time $Q(id)$ computation.

- Step 2: Verify the feasibility of the insertion

  For checking the insertion feasibility, the expressions are classified in three cases depending on the insertion positions. The three cases are:

  - the acquisition $Acq\ k$ can be inserted in the first position (position 1) of the selected acquisition set, only if

    $$Tmin(Acq\ k) + Du(Acq\ k) + Dt(Acq\ k, sa_1) \leq Q(1)$$

  - the acquisition $Acq\ k$ can be inserted in the middle position id (position 2 to N) of the selected acquisition set, only if

    $$ta_{id-1} + Du(sa_{id-1} + Dt(sa_{id-1}, Acq\ k)) \leq Tmax(Acq\ k)$$

and

$$Tk + Du(Acq\ k) + Dt(Acq\ k, sa_{id}) \leq Q(id)$$

where $Tk = max(Tmin(Acq\ \ k), ta_{id-1} + Du(sa_{id-1}) + Dt(sa_{id-1}, Acq\ k))$

– the acquisition $Acq\ k$ can be inserted in the last position (position $N + 1$) of the selected acquisition set, only if

$$ta_N + Du(sa_N) + Dt(sa_N, Acq\ k) \leq Tmax(Acq\ k)$$

- Step 3: If the acquisition $Acq\ k$ is a part of stereo request, its twin must also be checked for its insertion feasibility. The latest starting times from Step 1 are re-computed by using the updated selected acquisition set, which included $Acq\ k$ in the set.

  – If the twin of acquisition $Acq\ k$ cannot be inserted, the acquisition $Acq\ k$ has to be removed from the selected acquisition set.

This feasibility checking is used to verify the possibility of the neighbor generation for two main types of move, which are the insertion and replacement. If the considered neighbor is feasible, the selected acquisition set is modified and the neighbor is obtained. Then, the starting time set is re-generated depending on the modified selected acquisition set. The starting time of each acquisition is assigned as early as possible. The generated neighbor, which gives better fitness value than the best found neighbor, replaces the best found one. The possibilities of the other neighbors have to be checked until all neighbors in the neighborhood are explored.

The results obtained from these two methods of feasibility checking are compared in Chapter 5. The first method has an advantage that it has less number of steps in the process for checking the feasibility. But the second method has another advantage that it can increase the insertion possibility of each considered acquisition. Thus, the results will be compared both for the qualities of approximate Pareto front and the computation times.

5 Conclusion

This chapter presented the indicator-based multi-objective local search (IBMOLS) for solving the multi-user observation scheduling problem of an agile Earth observing satellite. The IBMOLS is a generic method, which combines the use of indicator-based evolutionary algorithm and the iterated local search for searching for the approximate Pareto front of the multi-objective optimization problem. The overview of IBMOLS, the initial generation, and the

fitness computation were explained. For the local search step, the neighborhood structure was presented. The insertion, removing, and replacement of the mono acquisition or the twin acquisitions of the stereo requests were proposed. All neighbors in the neighborhood are explored and the best neighbor is chosen. Before generating the neighbor, the feasibility has to be checked. Two methods are applied to verify the feasibility to generate the neighbor. Experiments will be presented in Chapter 5.

Computational results

5

1 Introduction

This chapter presents the computational results of the experiments for solving the multi-objective optimization problem of selecting and scheduling images taken by an agile Earth observing satellite. Two metaheuristic algorithms, consisting of the biased random key genetic algorithm (BRKGA) and the indicator-based multi-objective local search (IBMOLS), are used to solve the problem. The experiments are conducted on realistic instances, which were proposed in the context of ROADEF 2003 challenge. In our work, the problem was modified so as to consider requests explicitly emanating from several users. Hence, the ROADEF 2003 challenge instances (Subset A) (see `http://challenge.roadef.org/2003/en/sujet.php`) are modified for 4-user requirements and the format of instance names are changed to `a_b_c`, where `a` is the number of requests, `b` is the number of stereo requests, and `c` is the number of strips.

This chapter is organized as follows. Section 2 presents the results obtained by applying BRKGA. This section proposes and compares the results in three parts. The first part reports the results obtained with two single decoding methods. They are a basic decoding and a decoding of gene value with ideal priority combination. The elite solutions are selected by using three methods, which are borrowed from efficient multi-objective evolutionary algorithms. The second part presents the results obtained by using a hybrid decoding method. In this part, an issue of the hybrid decoding, that two feasible solutions are obtained from the decoding of one chromosome, are solved by using elite set management methods; their respective experimental results are compared. The last part compares the results from the two single decodings and the results from the hybrid decoding regarding the hypervolume values

of the approximate Pareto front and the average computation times. Section 3 presents the results obtained with IBMOLS. Several parameter settings are studied and the respective results are compared. This section is presented in four parts. In the first part, different strategies are used to generate the initial population. In the second part, two groups of neighborhood structure are defined. One group includes the replacement strategy, but the other one does not. For the third part, the results from two methods for feasibility checking are compared. The first one checks the insertion feasibility directly with the original starting time set. The second method computes the latest starting times of the acquisitions, which are scheduled after the insertion position before checking the feasibility. The last part presents the results obtained from different stopping criteria in the IBMOLS process. In Section 4, the best results obtained from BRKGA and IBMOLS are compared.

2 Biased random key genetic algorithm

Our implementation of BRKGA was described in Chapter 3. The experiments are done on the modified ROADEF 2003 challenge instances (Subset A). The obtained results are reported and discussed in this section.

For the proposed biased random key genetic algorithm (BRKGA), parameter values of the algorithm were experimentally tuned for this work. The population size p of BRKGA is set equal to the length of the random key chromosome or twice the number of strips. As presented in Chapter 3, the population for the next iteration in BRKGA process is generated by composing three sets of chromosomes. They are the elite set, the crossover offspring set, and the mutant set. The sizes of the three sets are set in accordance with the recommended values in Table 1.1. The size of the elite set is equal to the number of non-repeating schedules from the nondominated solutions, but it is not over $0.15p$. The size of mutant set is equal to $0.3p$. The probability of elite element inheritance for crossover operation is set to 0.6. In each iteration, the potentially efficient solutions are stored in an archive. If there is at least one solution from the current population that can dominate some solutions in the archive, the archive will be updated. Therefore, we use the number of iterations since the last archive improvement as a stopping criterion. We opt for 50 iterations. Moreover, the computation time is used as the second stopping criterion. It is adapted to the instance size, as shown in Table 5.1. BRKGA will be stopped, when one of the two stopping criteria is satisfied.

The algorithm is implemented in C++ and ten runs per instance are tested. The hypervolumes of the approximate Pareto front are calculated by using a reference point of 0 for the first objective (maximizing the total profit) and the maximum of the profit summations of each user for the second objective (minimizing the maximum profit difference between users). The hypervolume

Table 5.1: The BRKGA limitation values of computation time for the modified ROADEF 2003 challenge instances (Subset A).

Instance	Limit computation time value (seconds)
4_0_7	100
12_2_25	100
12_9_28	100
68_12_106	2000
77_40_147	2000
218_39_295	2000
150_87_342	10000
336_55_483	10000
375_63_534	10000

values of the results are plotted by the box plot. The results of the smallest instance (instance 2_0_2) cannot be reached, when using the population size equal to the length of the chromosome or twice of number of strips, because the population size is too small for generating the population in the next iteration from the three sets of chromosomes in BRKGA process. Thus, the results of nine instances, except the smallest one, will be reported in this section.

2.1 Basic decoding vs Decoding of gene value with ideal priority combination

Firstly, the experiments are done using the basic decoding (D1) and the decoding of gene value with ideal priority combination (D2). Three methods for elite set selection, which are borrowed from three efficient algorithms (NSGA-II, SMS-EMOA, and IBEA), are applied. In this section, the results of both decoding methods using the three methods of elite set selection are reported. The box plots of hypervolume values and the average computation times are shown in Figure 5.1.

There are six box plots on each graph. The first, third, and fifth columns show the results obtained with D1. The second, fourth, and sixth columns show the results obtained with D2. According to the presented results, the three methods for selection of the elite set (NSGA-II, SMS-EMOA, and IBEA) obtain similar box plots. The results on medium-size instances (12_9_28, 68_12_106, 77_40_147, and 218_39_295) show that the decoding of gene value with ideal priority combination (D2) obtains better results, both for the median values of hypervolumes and the standard deviations. Moreover, it spends less computation time than the basic decoding

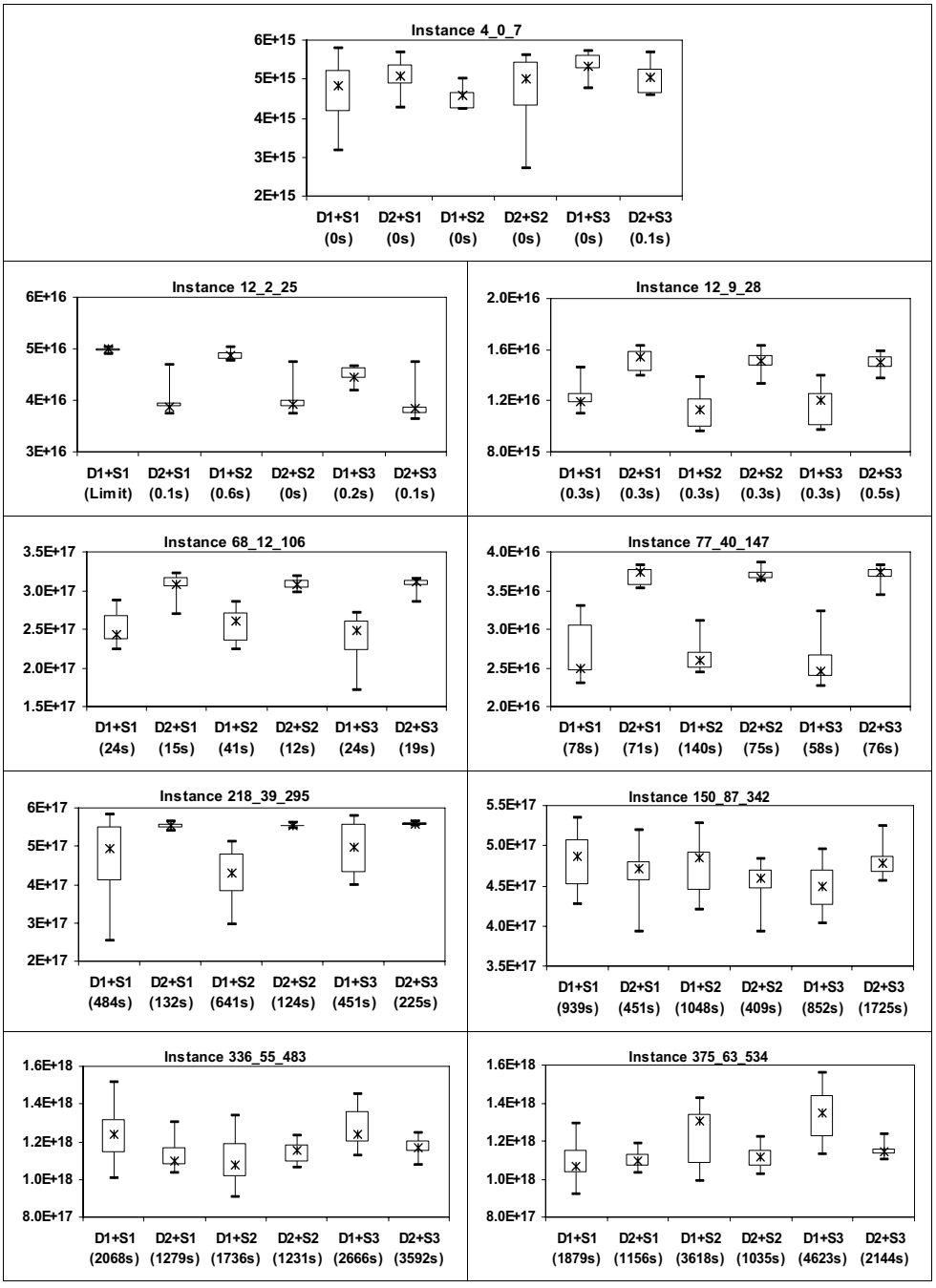


Figure 5.1: Comparison of the hypervolume values of the approximate Pareto front and the average computation times between two decoding methods: the basic decoding (D1) and the decoding of gene value with ideal priority combination (D2), by using the three methods for elite set selection borrowed from: NSGA-II (S1), SMS-EMOA (S2), and IBEA (S3).

(D1). For large-size instances, the range of the hypervolume values can be reduced with D2, but D1 obtains better median values of hypervolumes.

The results show that each decoding method has its own advantages. Hence, the idea of hybrid decoding is proposed in order to combine two single decoding methods and try to preserve the advantage of each one. However, the elite set management is an issue of the hybrid decoding, because two solutions will be obtained from the decoding of one chromosome. The next section will report the results, which are obtained from three different methods for elite set management. The three methods have been explained in Section 3.2 of Chapter 3.

2.2 Elite set management for hybrid decoding

With a hybrid decoding, a chromosome can be decoded to become two solutions. Thus, the methods to manage the elite set must be defined. Three methods are tested. The first method (M1) compares the two solutions obtained from the decoding of a chromosome. The solution that dominates the other one is selected to be stored in the solution set. The second method (M2) stores both solutions in a unique solution set. The third method (M3) stores the solution from each decoding method in two separate solution sets. For the three methods, the hypervolume values of the approximate Pareto front are computed and plotted in box plots. The box plots associated with the mechanisms of NSGA-II (S1), SMS-EMOA (S2), and IBEA (S3), are reported in Figures 5.2, 5.3, and 5.4, respectively. Each graph contains three columns corresponding to M1, M2, and M3.

The results show that the three methods obtain similar solutions regarding the hypervolume values of the approximate Pareto front. However, M2 spends more computation time for large instances, especially when the elite set selection method borrowed from IBEA is used. Furthermore, M3 spends more computation time for small instances, particularly with the elite set selection methods borrowed from NSGA-II or SMS-EMOA. Therefore, in the sequel only method M1 will be kept to compare the results between the hybrid decoding and the two single decoding approaches.

2.3 Decoding methods

In this section, three decoding methods, namely the basic decoding (D1), the decoding of gene value with the ideal priority combination (D2), and the hybrid decoding (HD), are compared. The box plots from the three elite set selection methods coming from NSGA-II, SMS-EMOA, and IBEA, are reported in Figures 5.5, 5.6, and 5.7, respectively. As previously, the graphs

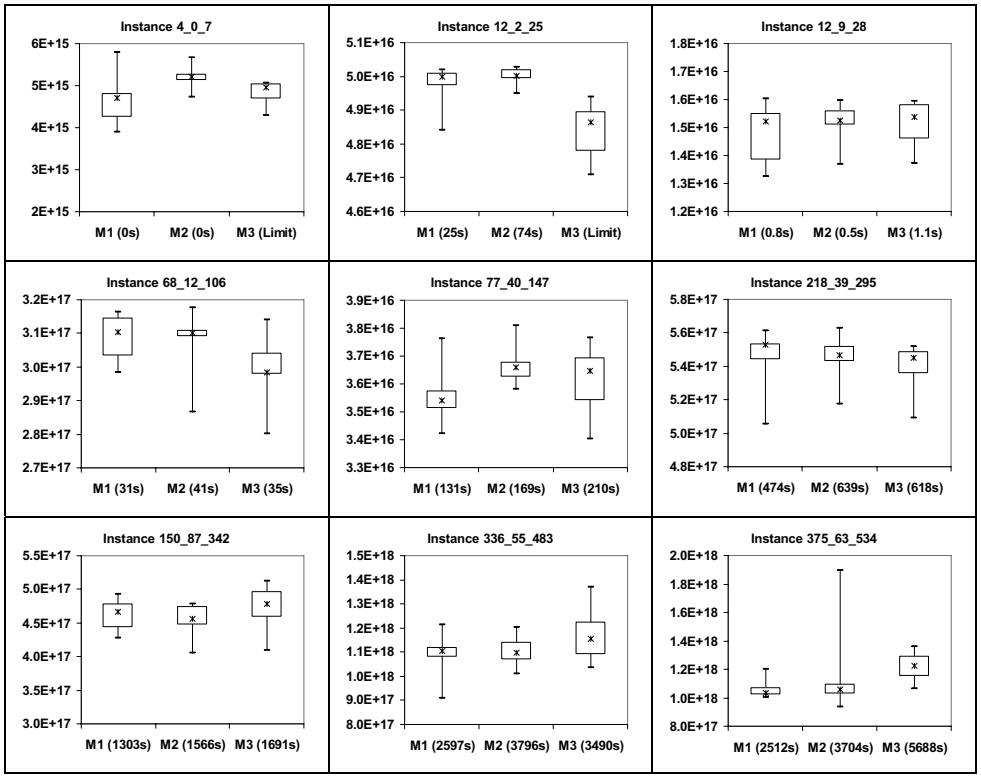


Figure 5.2: Comparison of the hypervolume values of the approximate Pareto front and the average computation times between three methods for management of the elite set for hybrid decoding: select dominant solution (M1), combine all solutions in a solution set (M2), and store the solutions from each decoding in two separate solution sets (M3), by using the method for elite set selection borrowed from NSGA-II (S1).

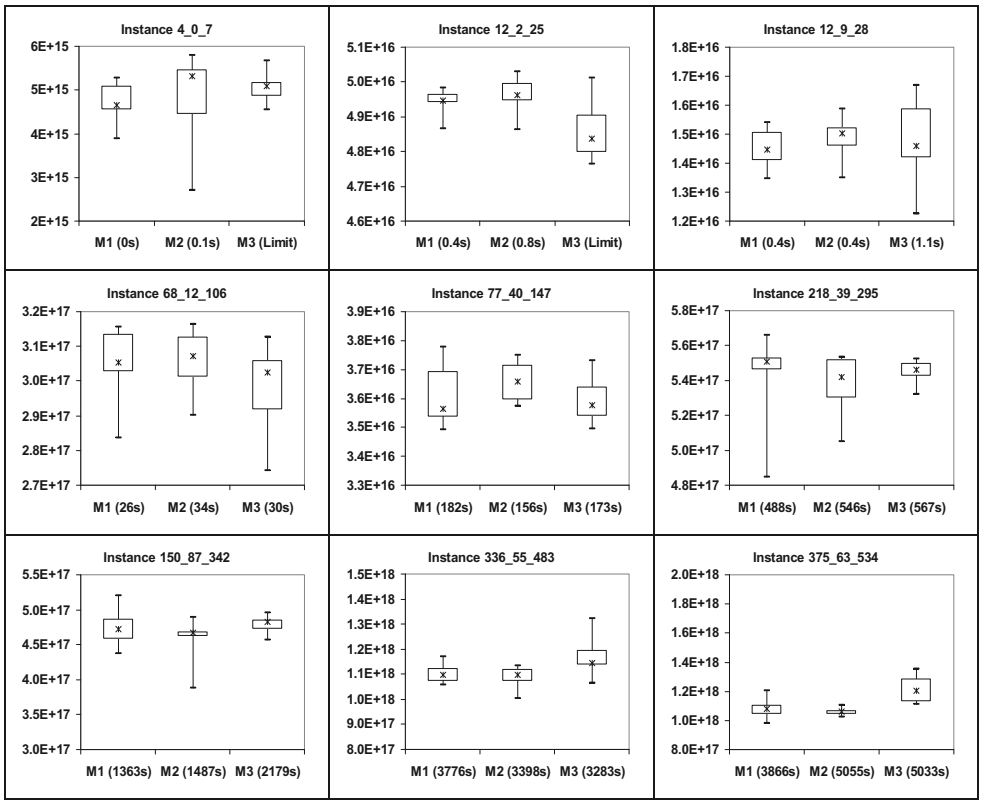


Figure 5.3: Comparison of the hypervolume values of the approximate Pareto front and the average computation times between three methods for management of the elite set for hybrid decoding: select dominant solution (M1), combine all solutions in a solution set (M2), and store the solutions from each decoding in two separate solution sets (M3), by using the method for elite set selection borrowed from SMS-EMOA (S2).

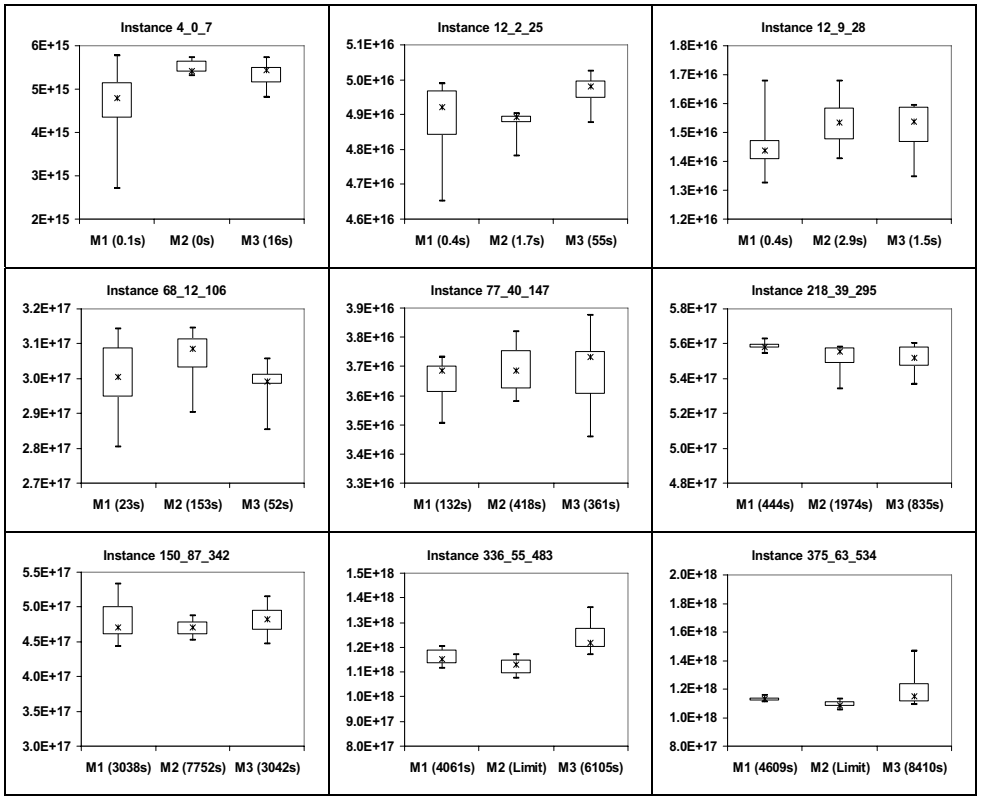


Figure 5.4: Comparison of the hypervolume values of the approximate Pareto front and the average computation times between three methods for management of the elite set for hybrid decoding: select dominant solution (M1), combine all solutions in a solution set (M2), and store the solutions from each decoding in two separate solution sets (M3), by using the method for elite set selection borrowed from IBEA (S3).

illustrate the box plots of the hypervolume values and the average computation times.

Most of the results show that the hybrid decoding obtains results close to the best ones, when comparing the two single decodings. Indeed, it can preserve the advantages of the two single decodings for all instances. For example, in instance `12_2_25`, the first decoding method obtains better results than the second one, thus the hybrid decoding obtains results similar to the first one. For instance `77_40_147`, the hybrid decoding obtains results similar to the second decoding, which obtains better results than the first one. Thus, the hybrid decoding method is efficient for solving most of the instances. Compared with D1, it can reduce the range of hypervolume values. This means that the hybrid decoding can provide results with better standard deviations. Moreover, for some instances where the second decoding entraps in local optima, the hybrid decoding is able to reach the global optimum. Regarding the computation time, the hybrid decoding method spends longer time in each iteration, however it can obtain good solutions in a reasonable computation time, which is limited by the second stopping criterion of BRKGA process.

3 Indicator-based multi-objective local search

This section presents the results, which are obtained from the indicator-based multi-objective local search (IBMOLS). The IBMOLS is a population-based method, which combines the uses of indicator-based evolutionary algorithm and the iterated local search. The parameter values of population size and stopping values have to be tuned by the experiments. In Sections 3.1, 3.2, and 3.3, a dynamic stopping criterion is applied. It corresponds to the number of iterations, without any improvement of the approximate Pareto front. The population size is set equal to 10. The algorithm is implemented in C++ and ten runs per instance are tested. The indicator-based fitness assignment with the hypervolume concept from IBEA is applied for comparing the solutions in the population. Hypervolumes of the approximate Pareto front are calculated by using a reference point of 0 for the first objective (maximizing the total profit) and the maximum of the profit summations of each user for the second objective (minimizing the maximum profit difference between users).

3.1 Initial population generation

Firstly, the dynamic stopping criterion is used in this section. The stopping value is set equal to 10. The methods for generating the initial population, which have been described in Sections 2.2 and 2.3 of Chapter 4, are tested.

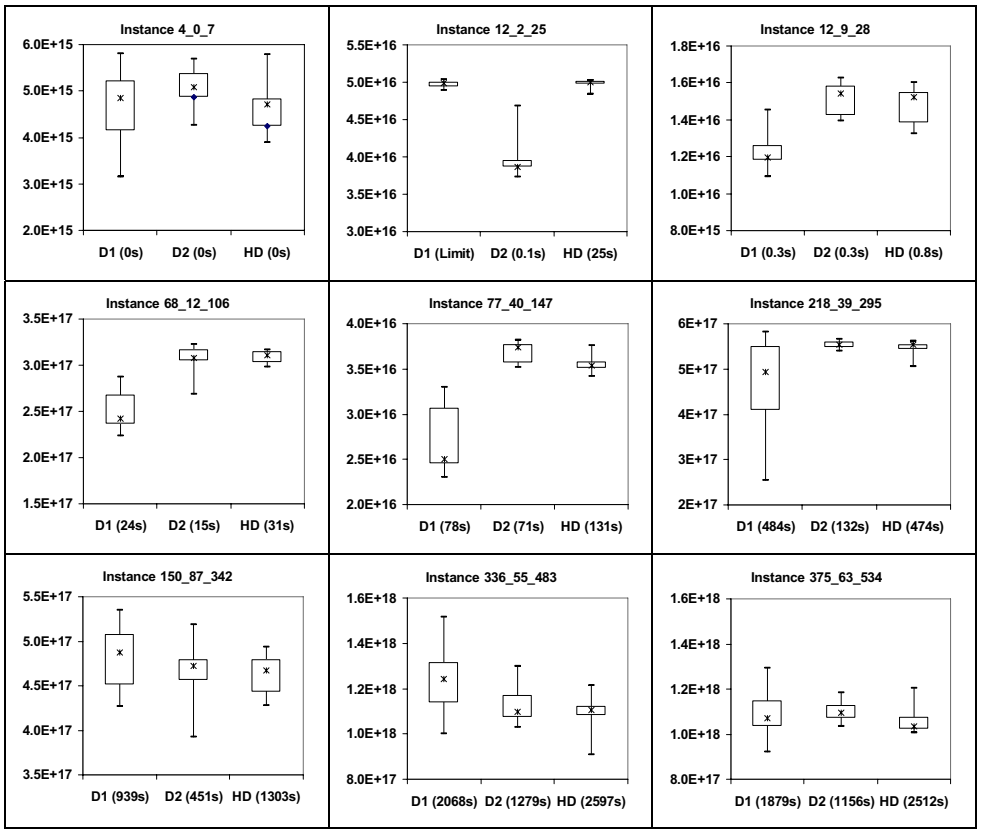


Figure 5.5: Comparison of the hypervolume values of the approximate Pareto front and the average computation times between three decoding methods: basic decoding (D1), decoding of gene value with the ideal priority combination (D2), and hybrid decoding (HD), by using the method for elite set selection borrowed from NSGA-II (S1).

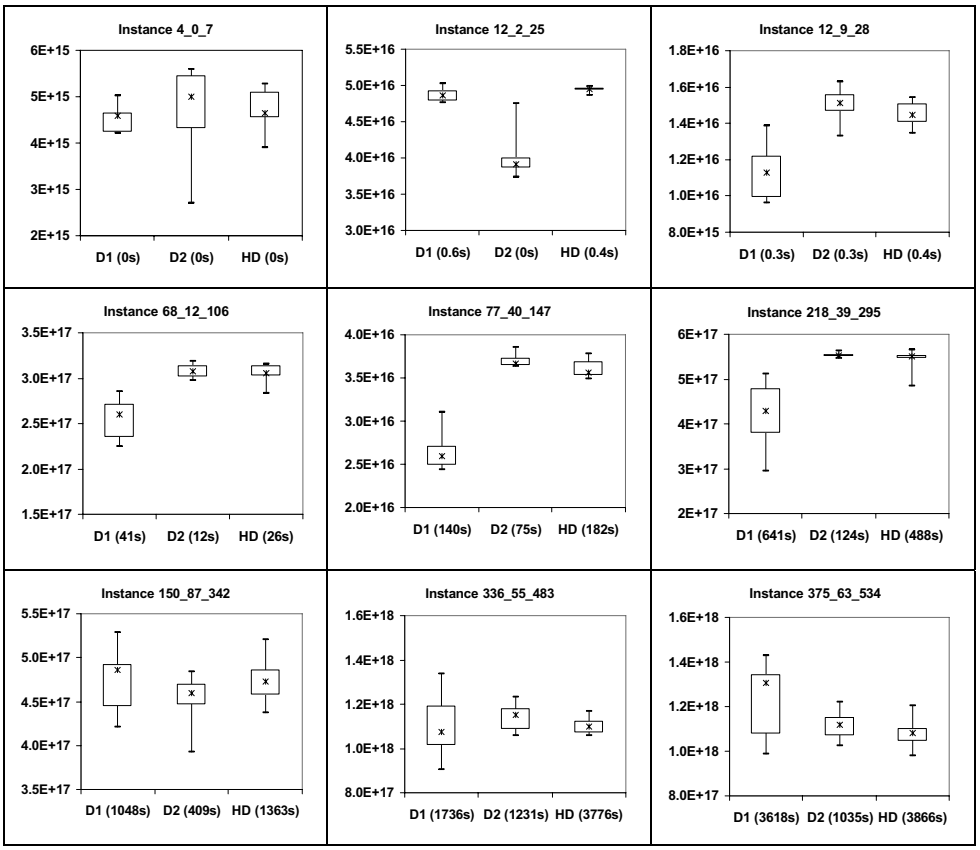


Figure 5.6: Comparison of the hypervolume values of the approximate Pareto front and the average computation times between three decoding methods: basic decoding (D1), decoding of gene value with the ideal priority combination (D2), and hybrid decoding (HD), by using the method for elite set selection borrowed from SMS-EMOA (S2).

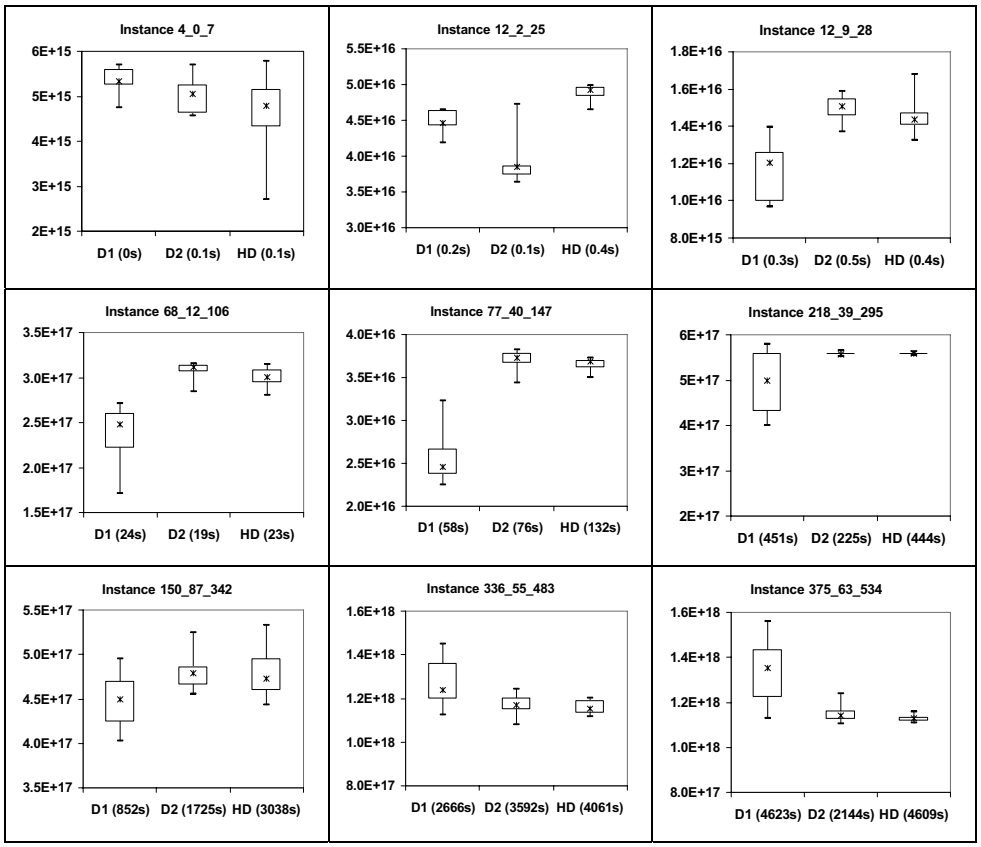


Figure 5.7: Comparison of the hypervolume values of the approximate Pareto front and the average computation times between three decoding methods: basic decoding (D1), decoding of gene value with the ideal priority combination (D2), and hybrid decoding (HD), by using the method for elite set selection borrowed from IBEA (S3).

The results obtained from three cases of initial population generation are compared. The first results are obtained by using the random generation for all iterations. The second results are obtained by using a random generation for the first iteration and using the perturbation for the other iterations. The third results are obtained by using the useful data of problem instances for generating the initial population in the first iteration and using the perturbation for the other iterations. The neighborhood structure, which consists of insertion, removing, and replacement of the mono and stereo acquisitions, is used in this experiment. Hypervolume values of the approximate Pareto front are calculated and they are plotted in box plots. The box plots of the three methods for generating the initial population including the average computation times are illustrated in Figure 5.8.

The results show that the perturbation, which is used to generate the initial population for the other iterations, obtains better results than using the random generation. For the first iteration, the random generation and the generation by using the useful data of problem instances obtains similar results and computation times. But the generation by using the useful data of problem instances needs more additional processes in order to generate the initial population than using the random generation. Thus, the random generation has the advantage for generating the initial population in the first iteration. In the next sections, the experiments will use the random generation in the first iteration and using the perturbation in the other iterations for generating the initial population in IBMOLS process.

3.2 Neighborhood structure

Secondly, two groups of neighborhood structures, which have been presented in Section 3 in Chapter 4, are tested. The first group applies six types of move, which are the insertion, removing, and replacement of the mono and stereo acquisitions. The second group applies four types of moves, which are the insertion and removing of the mono and stereo acquisitions. In this experiment, the setting values of the dynamic stopping criterion are also tested with the second group of the neighborhood structure. The hypervolume values of the approximate Pareto set are plotted in the box plots. The box plots and the average computation times are shown in Figure 5.9. In each graph, the first and second columns illustrate the results by using the stopping value equal to 10, but they are obtained from the first group (apply six types of move) and the second group (apply four types of move) of the neighborhood structure, respectively. The third column shows the results, which are obtained from the second group (apply four types of move) of the neighborhood structure and the stopping value equal to 50 is used.

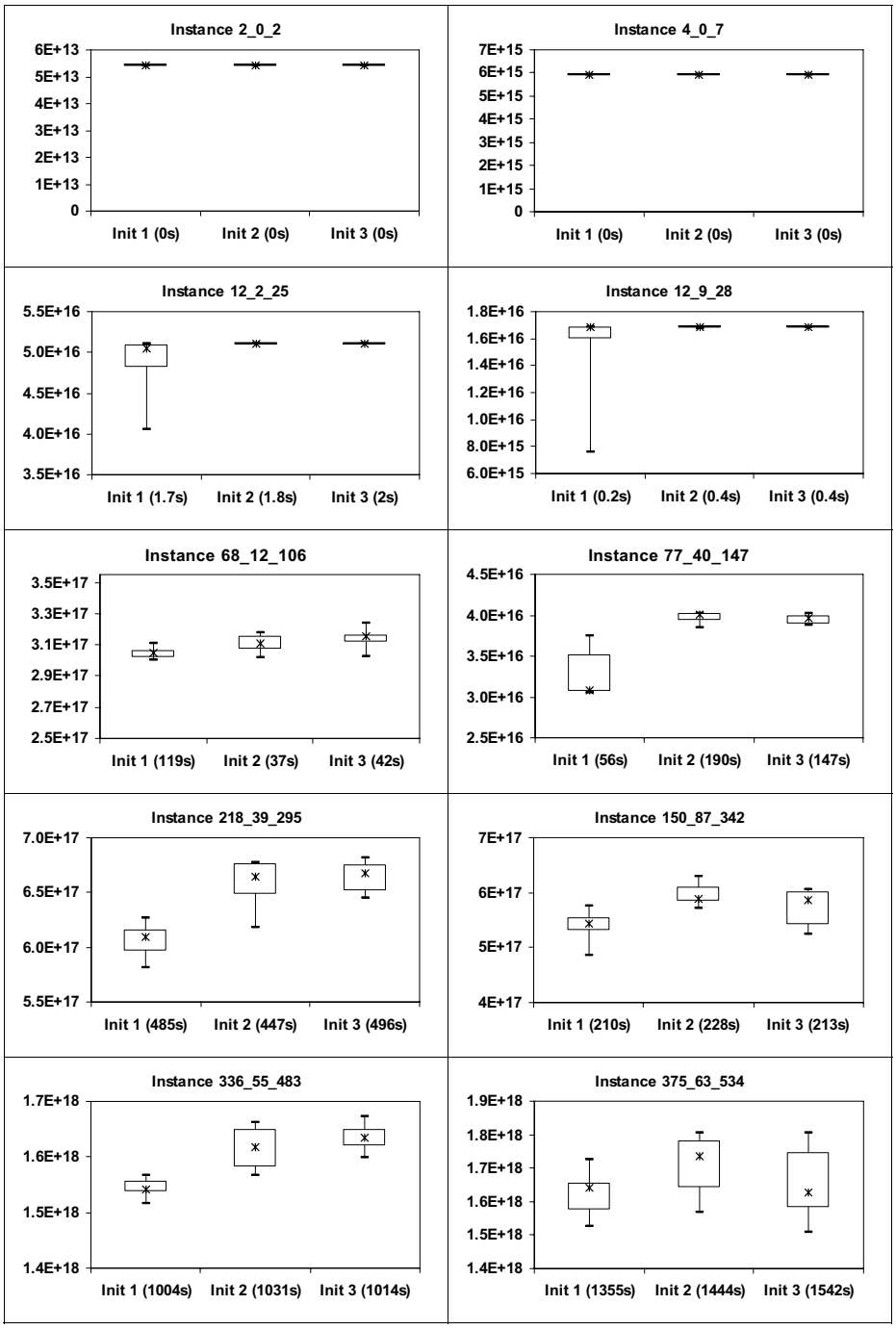


Figure 5.8: Comparison of the hypervolume values of the approximate Pareto front and the average computation times between three methods for generating the initial population for IBMOLS.

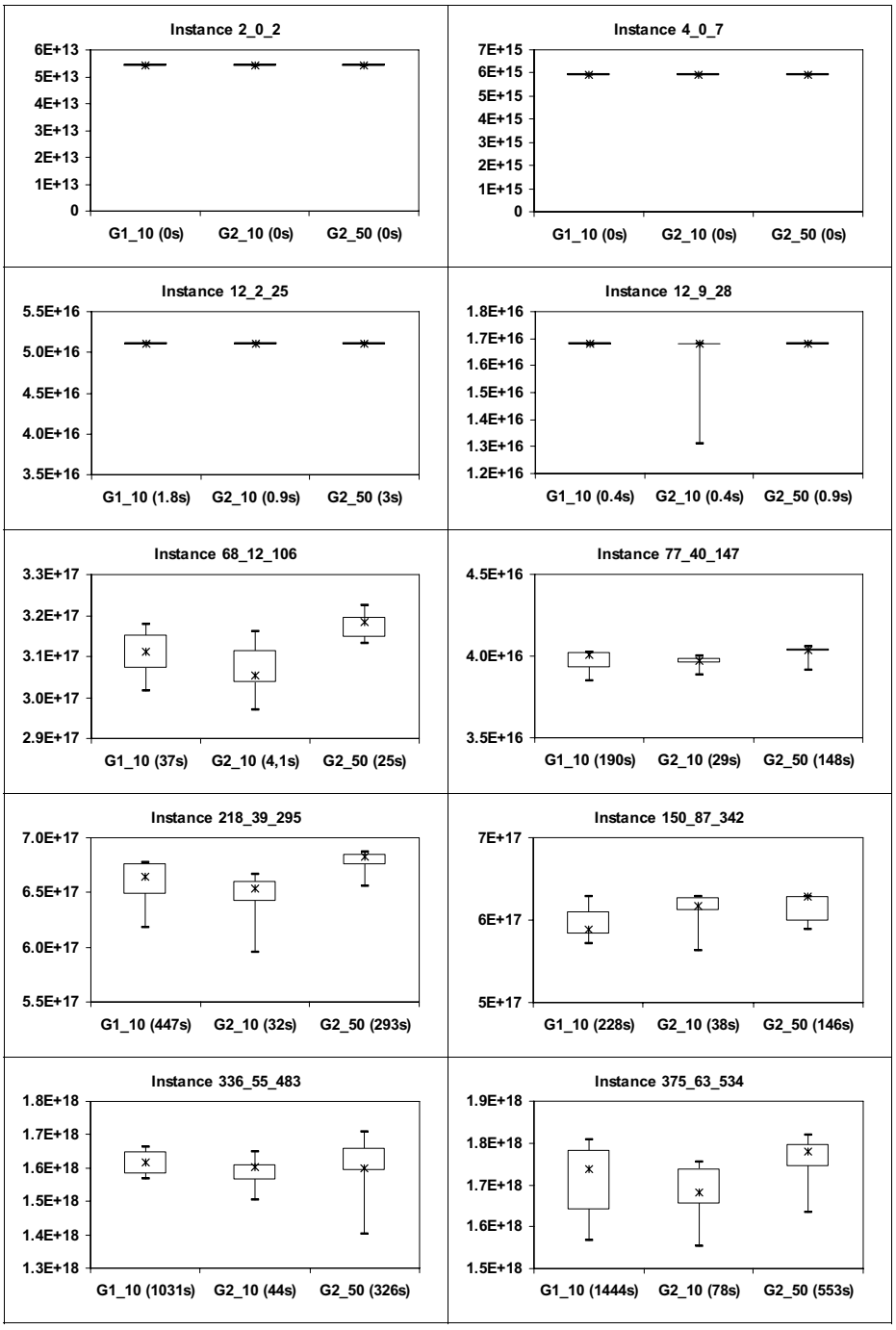


Figure 5.9: Comparison of the hypervolume values of the approximate Pareto front and the average computation times between two groups of neighborhood structure and two stopping values of dynamic stopping criterion for IBMOLS.

When the same value of dynamic stopping criterion is used and two groups of neighborhood structure are compared, most of the results show that the second group obtains results, which are a little bit worse than the first group. But the second group of neighborhood structure spends much less computation time. When the stopping value of the dynamic stopping criterion is increased, the obtained results can be improved and more computation time is spent. However, for most instances, except the small size ones, when using the second group of neighborhood structure and setting higher stopping value, the obtained hypervolume values and the consumptions of computation times are the best for this comparison.

3.3 Feasibility checking

Thirdly, two methods for checking the feasibility of the neighbors, which are presented in Section 4 of Chapter 4, are compared. The first method checks directly the feasibility to insert an acquisition with the starting time set of the original solution. The second method computes the latest starting time of the acquisitions, which stay behind the insert position, before checking the feasibility. It is sure that the second method should obtain better results, because it gives more possibility to insert the considered acquisition. Thus, in Sections 3.1 and 3.2, the presented results are obtained from the second method of the feasibility checking. This section wants to illustrate the difference between the two methods for feasibility checking. The results, which are obtained by using the first and second methods, are shown in Figure 5.10.

In this comparison, the stopping value is set equal to 10 and the first group of neighborhood structure is applied. The results show that, the second method for checking the insertion feasibility obtains better results than the first one. However, the second method spends more computation time. Indeed, it has to calculate the latest starting time of the acquisitions, which stay behind the insertion position, before checking the insertion feasibility.

3.4 Stopping criterion

In the previous parts, the presented results are obtained by using a dynamic stopping criterion. In this part of IBMOLS experiments, other stopping criteria are tested. They are a fixed computation time and a fixed number of visited neighbors. The values of the second and the third stopping criteria are specified for each instance as in Tables 5.2 and 5.3, respectively. Moreover, the two groups of neighborhood structures are tested for all instances. The results, which are obtained by using the computation time as the stopping criterion, are presented in Figure 5.11. The first and second columns of each graph show the results, which are obtained from the first group (apply six types of

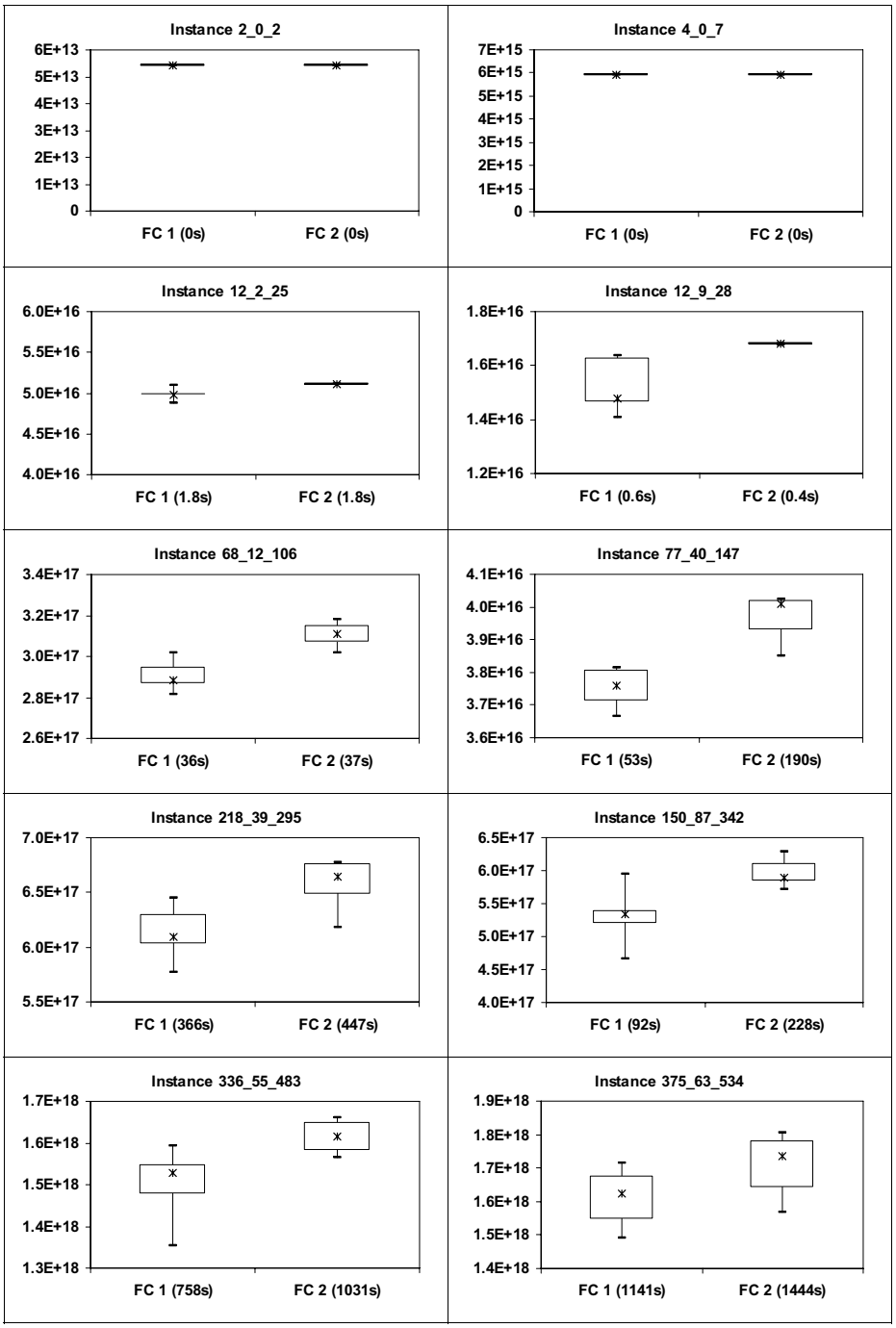


Figure 5.10: Comparison of the hypervolume values of the approximate Pareto front and the average computation times between two methods of feasibility checking.

Table 5.2: Stopping values concerning the computation time in the experiments of IBMOLS for the modified ROADEF 2003 challenge instances (Subset A).

Instance	Stopping computation time value (seconds)
2_0_2	5
4_0_7	5
12_2_25	5
12_9_28	5
68_12_106	300
77_40_147	300
218_39_295	300
150_87_342	300
336_55_483	1000
375_63_534	1000

move) and the second group (apply four types of move) of the neighborhood structure. Similarly, the results, which are obtained by using the number of visited neighbors as the stopping criterion, are illustrated in Figure 5.12.

When the computation times are fixed, the second group of neighborhood structure obtains better results for most of the instances. For the other instances, the obtained results are similar between the first and the second groups of neighborhood structure. When the fixed numbers of the visited neighbors are used as the stopping criterion, the first group of neighborhood structure obtains better results, because of the replacement process, which is not operated in the second group of neighborhood structure. For the first group, more various neighbors are visited in one iteration and the results can be improved by visiting a fewer number of neighbors. However, the first method of neighborhood structure spends more computation time to check and generate a neighbor in the replacement process of the neighborhood structure.

4 BRKGA vs IBMOLS

In this section, the results of the biased random key genetic algorithm (BRKGA), presented in Section 2, and the results of the indicator-based multi-objective local search (IBMOLS), presented in Section 3, are compared. For each algorithm, thirty runs per instance are tested. For BRKGA, the compared results are obtained from the hybrid decoding, which uses the selection method of indicator-based evolutionary algorithm (IBEA)

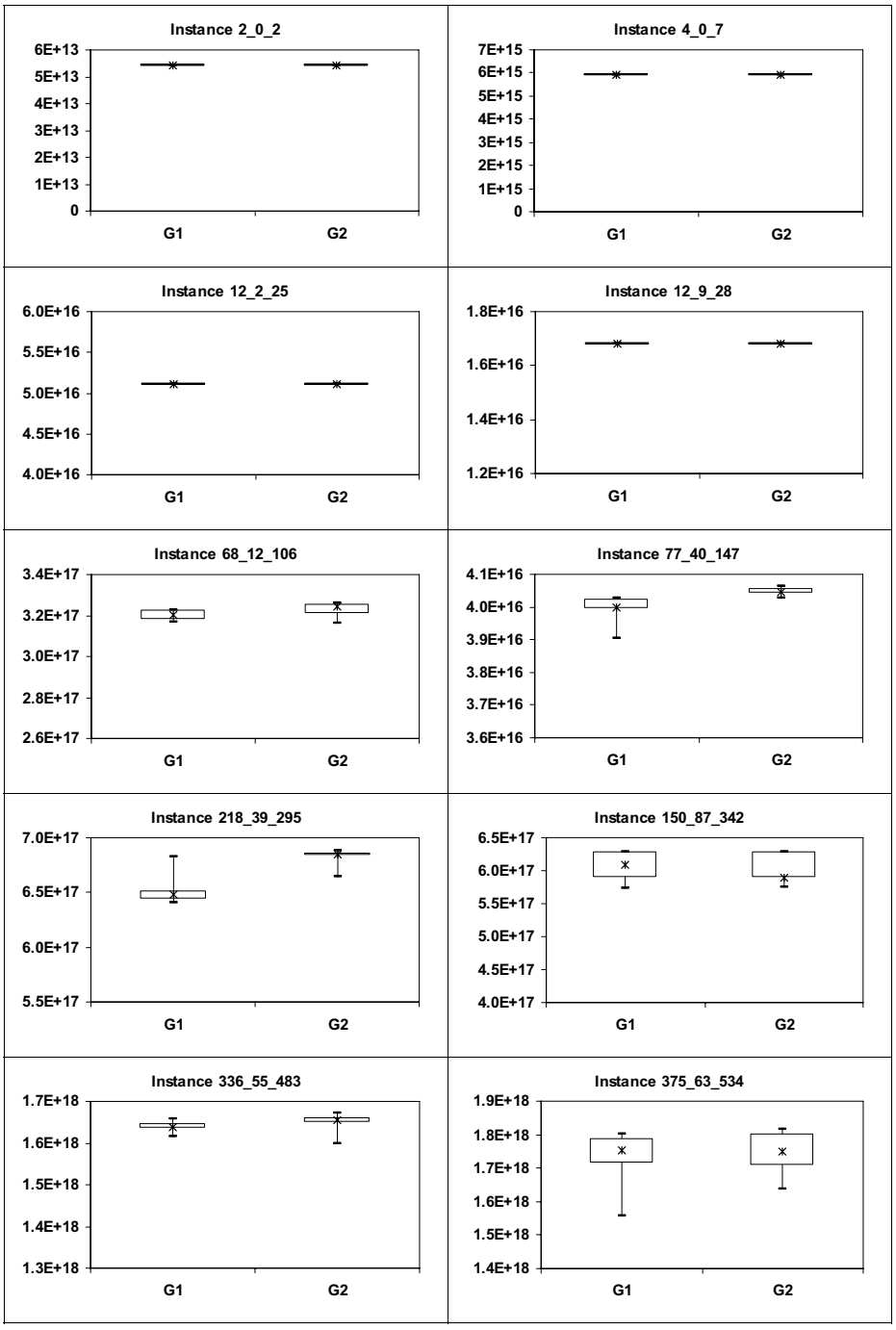


Figure 5.11: Comparison of the hypervolume values of the approximate Pareto front between two groups of neighborhood structure by using the computation time as the stopping criterion.

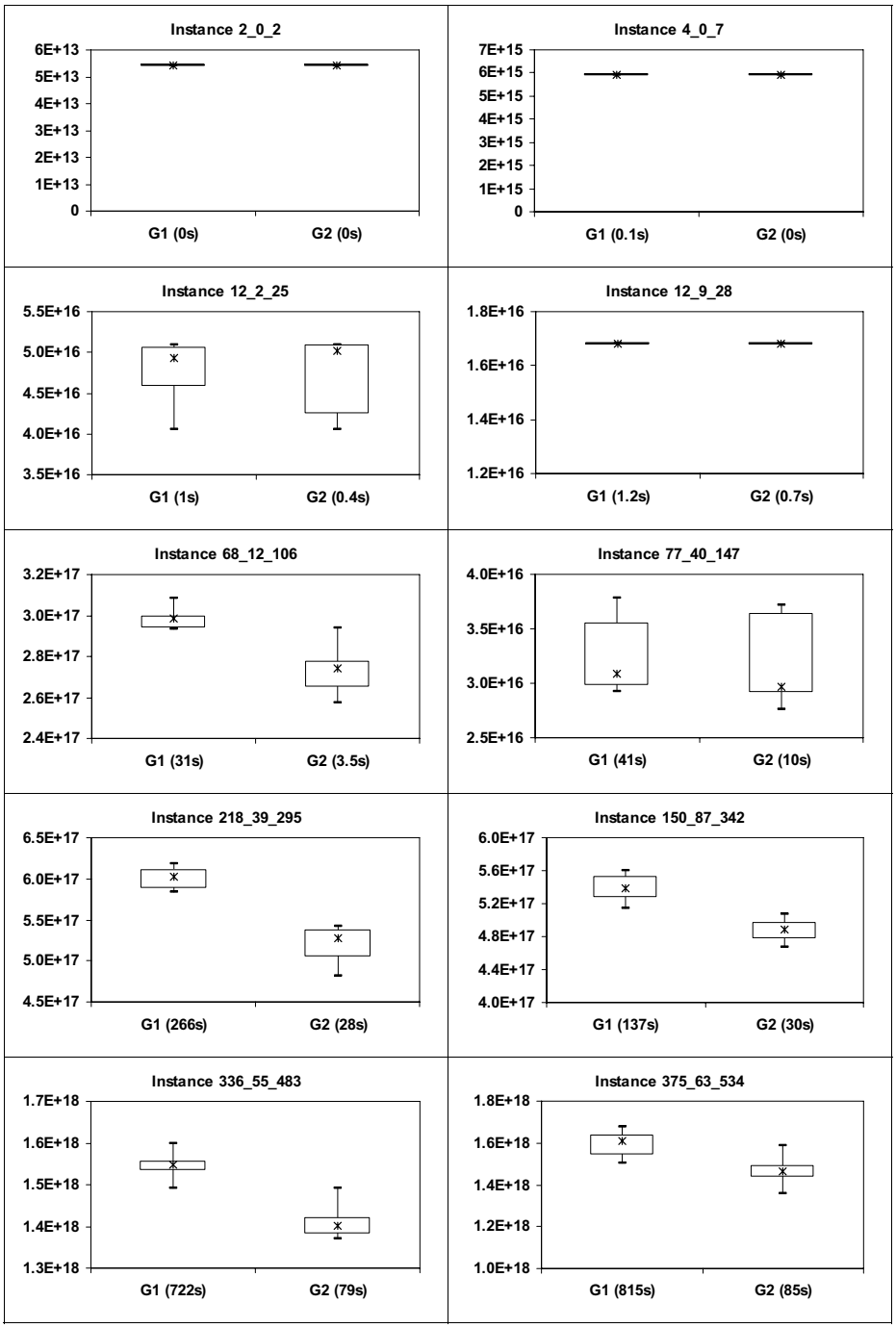


Figure 5.12: Comparison of the hypervolume values of the approximate Pareto front and the average computation times between two groups of neighborhood structure by using the number of visited neighbors as the stopping criterion.

Table 5.3: Stopping values concerning the number of visited neighbors in the experiments of IBMOLS for the modified ROADEF 2003 challenge instances (Subset A).

Instance	Stopping value of the number of visited neighbors
2_0_2	0
4_0_7	1000
12_2_25	10000
12_9_28	10000
68_12_106	20000
77_40_147	40000
218_39_295	40000
150_87_342	40000
336_55_483	60000
375_63_534	60000

for selecting the preferred chromosomes to become the elite set. The first method of the elite set management, which selects the dominant solution between the two decoding, is used to obtain the results in this comparison. For IBMOLS, the compared results are obtained by using the neighborhood structure, which consists of the insertion and removing of the mono and stereo acquisitions. The second method of feasibility checking, which calculates the latest starting time before checking insertion feasibility, is applied. For generating the initial population, the random generation is used in the first iteration and the perturbation is applied in the other iterations. The number of iterations of the last archive improvement is used as a stopping criterion for both BRKGA and IBMOLS. The stopping value is set to 50. The box plots of hypervolume values and the average computation time of BRKGA and IBMOLS are presented in Figure 5.13. The first column illustrates the results from BRKGA and the second column shows the results from IBMOLS. Moreover, we also use a Mann-Whitney statistical test for comparing the results from both algorithms.

The box plots show that IBMOLS obtains better median values of the hypervolume for all instances and better standard deviations for most of the results. Moreover, IBMOLS spends less computation time than BRKGA, especially for the large instances. Additionally, the results of IBMOLS are significantly better than those of BRKGA. In Figure 5.14, the improvement of the hypervolume values versus the computation times of the medium and the large instances are illustrated. In each graph, the improvement of hypervolume values between BRKGA process and IBMOLS process are compared.

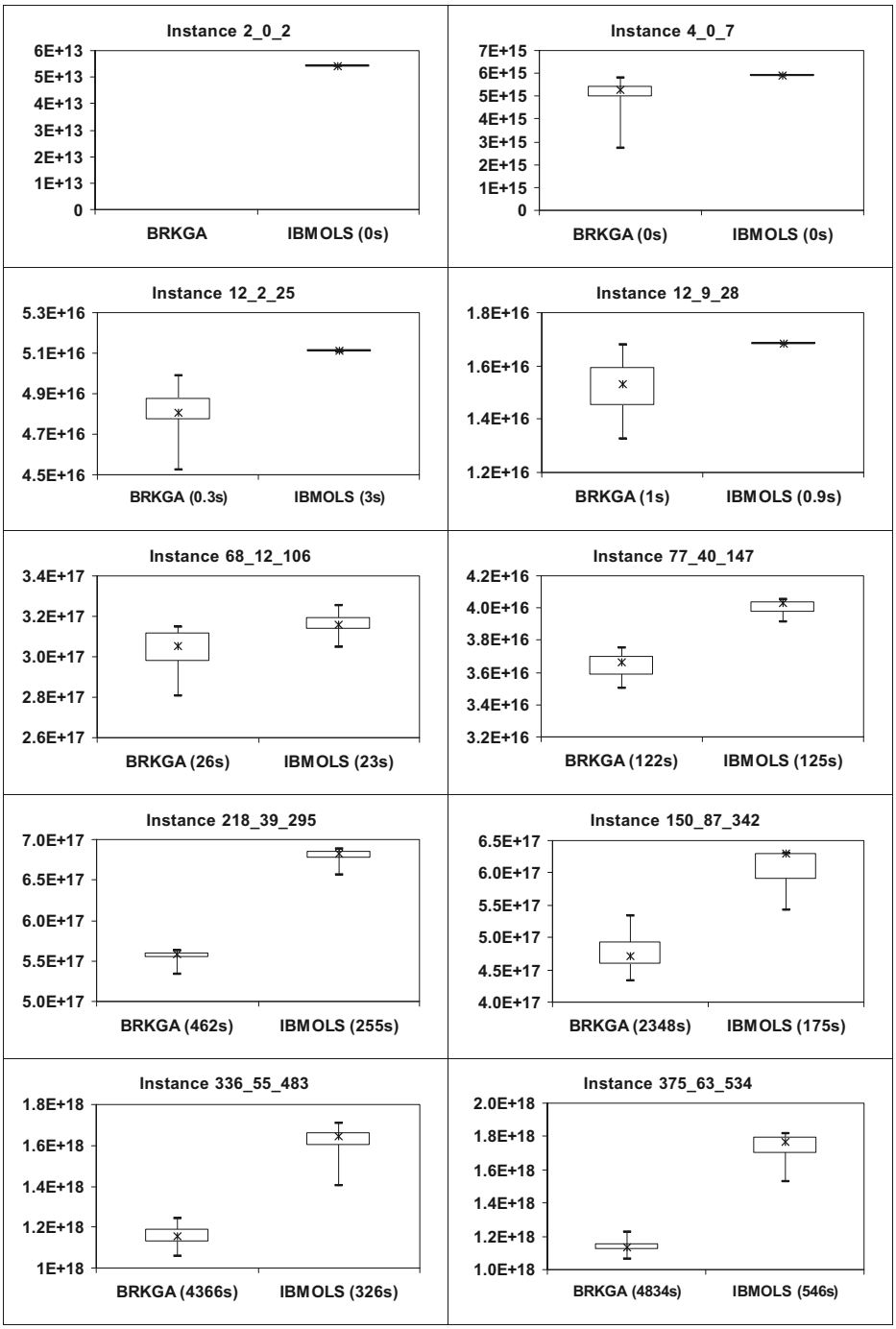


Figure 5.13: Comparison of hypervolume values of the approximate Pareto front and the average computation times between the best results from BRKGA and the best results from IBMOLS.

The results show that IBMOLS obtains solutions closer to the global optimum for the medium and large instances. Moreover, it can also converge to the global optimum faster than BRKGA.

Finally, the best approximate Pareto fronts of all instances are illustrated in Figure 5.15. For each instance, the total profit is presented on x-axis and the maximum profit difference between users is presented on y-axis.

5 Conclusion

This chapter presents the computational results for the multi-objective optimization problem under study (observation scheduling of an agile observing satellite). Two algorithms are applied to solve this problem. The first algorithm is the biased random key genetic algorithm (BRKGA). The selection methods of three established multi-objective evolutionary algorithms are used to select the preferred chromosomes. For the decoding step, a hybrid decoding is proposed. With the hybrid decoding, the obtained solutions are better than when using only one single decoding. The second algorithm is the indicator-based multi-objective local search or IBMOLS. This algorithm combines the uses of indicator-based evolutionary algorithm and the iterated local search. The best improvement strategy is used to select a neighbor in the neighborhood to replace the worst solutions in the population. Several methods for generating the initial population and two neighborhood structures are tested. IBMOLS obtains very good results for the considered problem. Finally, the best results, which are obtained from BRKGA and IBMOLS, are compared. IBMOLS reaches better results than BRKGA, both in median values and standard deviations for the hypervolume of the approximate Pareto front. Moreover, IBMOLS spends less computation times and it can converge to the global optimum faster than BRKGA, especially for the large instances.

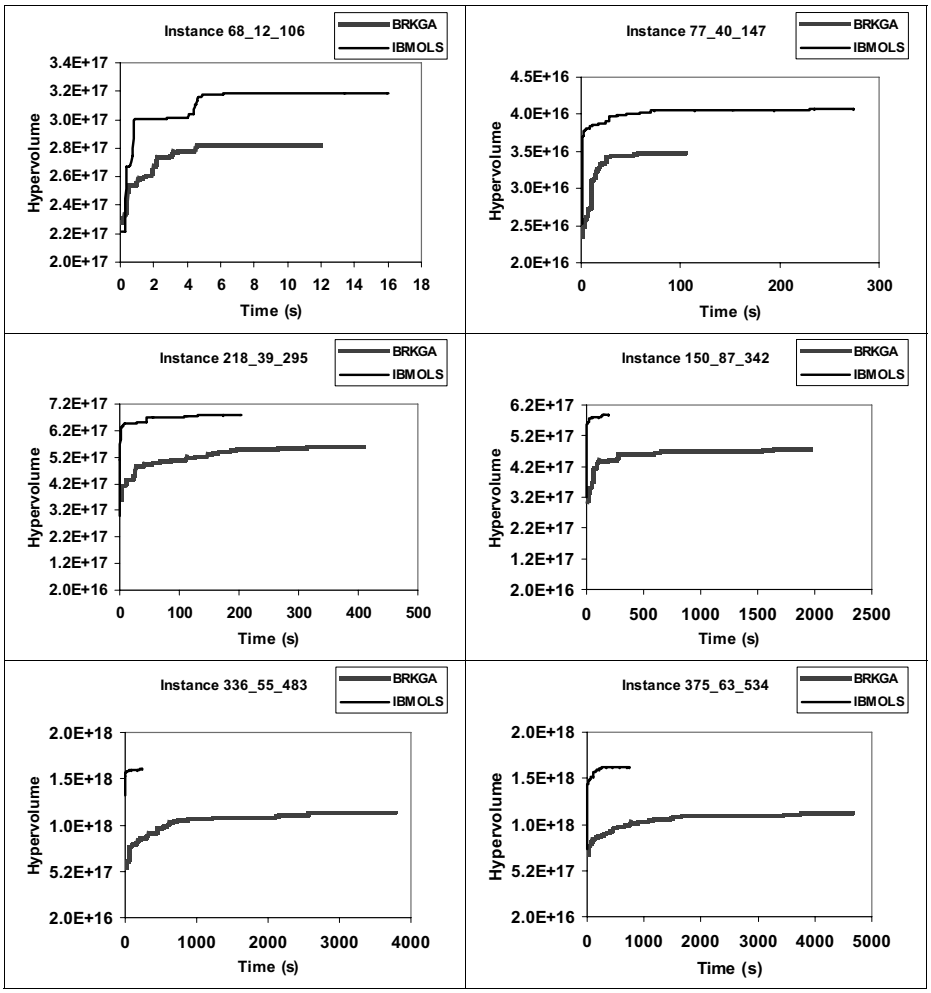


Figure 5.14: Comparison of the improvement of hypervolume values versus the computation times between BRKGA process and IBMOLS process when using the dynamic stopping as the stopping criterion.

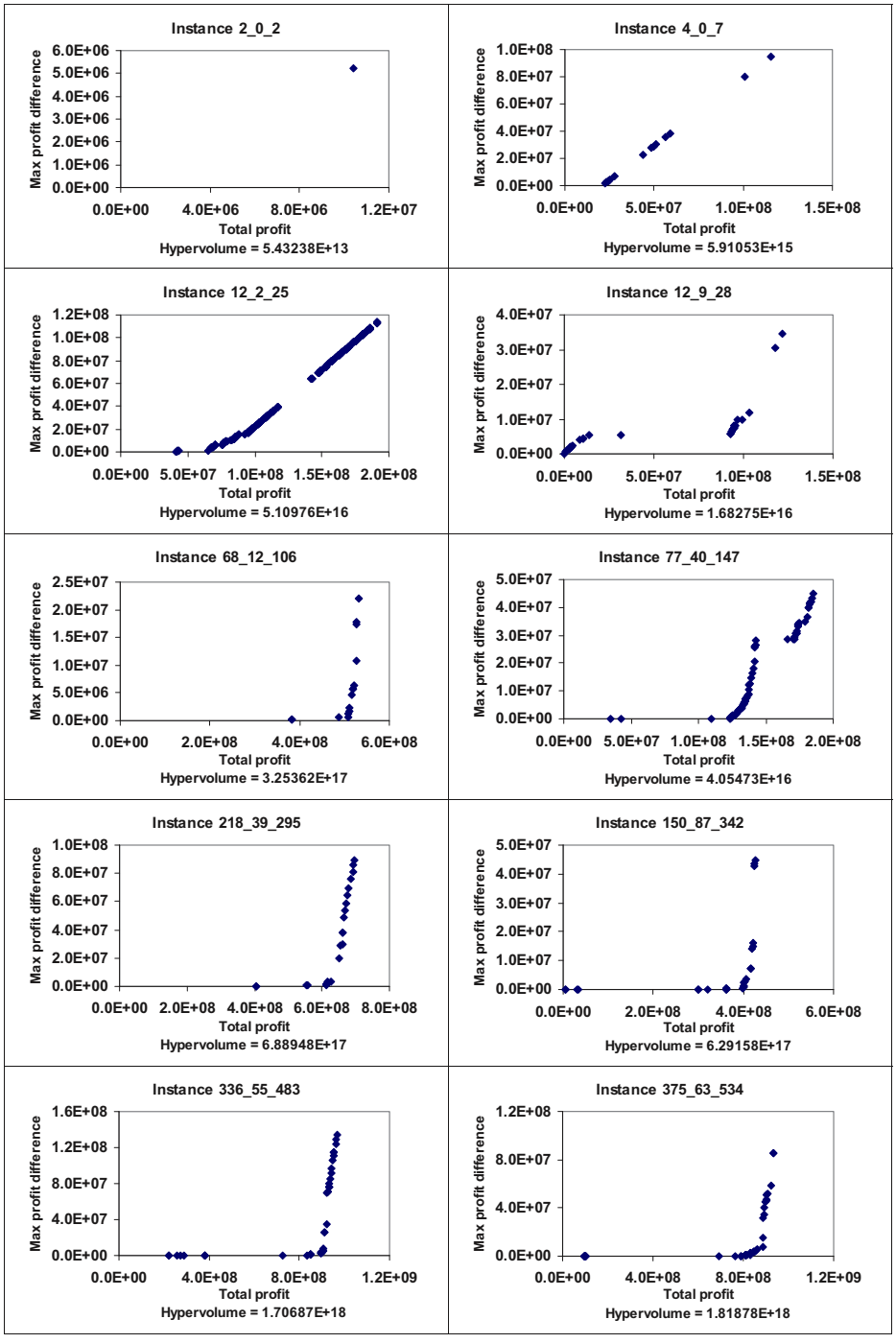


Figure 5.15: The best approximate Pareto front of each instance.

Conclusions and perspectives

The multi-user observation scheduling problem of an agile Earth observing satellite has been solved in this work. The problem considers the requests demanded from several users. These requests have to be managed before being transmitted to the satellite. The requests must be decomposed into several strips, which can be taken only once by the satellite. Each strip can be acquired from two opposite directions (called acquisitions), but only one direction can be chosen. If the acquisitions are selected to be scheduled in the sequence and acquired by the satellite, they can give profits and gains. Hence, one way for obtaining the highest profit is to assign as many acquisitions as possible in the sequence. However, in the real case, the satellite usually cannot acquire too many acquisitions in only one revolution, because of several reasons. For example, the number of acquisitions may exceed the satellite capacity, or more than one required area must be acquired during the same period.

The problem is modeled as a multi-objective optimization problem. Two objectives are optimized under operation constraints. The first objective is to maximize the total profit. The second one is to ensure the fairness of resource sharing, by minimizing the maximum profit difference between users. The operation constraints, which have to be satisfied, consist of time windows, sufficient transition times, one of two directions can be acquired for each strip, and stereo constraints. After solving this optimization problem, a sequence of selected acquisitions is obtained and then, it is transmitted in order to operate on the satellite. The ROADEF 2003 challenge instances (Subset A) are modified in order to consider the case of 4-user requirements. Experiments have been done on these instances.

Two metaheuristics, a biased random key genetic algorithm (BRKGA) and an indicator-based multi-objective local search (IBMOLS), are applied to solve this problem. For BRKGA, some steps are adapted for solving the multi-objective optimization problem. In the experiments, two stopping cri-

teria are used: a dynamic stopping, which is a number of iterations after the last improvement, and a limit computation time. Three methods for the elite set selection, which are borrowed from NSGA-II, SMS-EMOA, and IBEA, are applied to select a set of preferred chromosomes to become the elite set. Firstly, two single decoding methods, which consist of a basic decoding and a decoding of gene value with ideal priority combination, are used separately to decode chromosomes to be solutions. Each single decoding method has its own advantages. Hence, we propose the idea of hybrid decoding in order to combine two single decoding methods and try to preserve the advantage of each one. However, the elite set management is an issue of the hybrid decoding, because two solutions will be obtained from the decoding of one chromosome. Three methods, which are used to manage the elite set of hybrid decoding, are proposed. Then, the results from the hybrid decoding and the two single ones are compared. The hybrid decoding is efficient to provide results with better standard deviation than the basic decoding. Moreover, it can avoid the entrapment in a local optimum, which may happen when using the decoding of gene value with ideal priority combination. Hence, the hybrid decoding obtains better solutions than when using only one single decoding. Furthermore, the corresponding computation time is reasonable.

For IBMOLS, we use the best improvement strategy to select the neighbor in the neighborhood to replace the worst solution in the population. A dynamic stopping, which is a number of iterations after the last improvement, is used to be the stopping criterion. Three methods for the initial population generation are proposed. We obtain the best results when using the random generation for the first iteration and the perturbation for the other iterations. The perturbation can improve the solutions faster than using the random generation for the other iterations. However, for the first iteration, the random generation is enough for generating the initial population, with no need of an additional process. Moreover, the neighborhood structure and the feasibility checking are also studied. The neighborhood structure with the replacement obtains better solutions, but spends more computation time. For the feasibility checking, we propose to compute the latest starting time of the assigned acquisitions, before checking the insertion feasibility. IBMOLS obtains very good results for the considered problem. Furthermore, other stopping criteria, which are a fixed computation time and a fixed number of visited neighbors, are also presented.

The best results from BRKGA and IBMOLS are compared. IBMOLS reaches better results than BRKGA, both in median values and standard deviations for the hypervolume of the approximate Pareto front. Moreover, IBMOLS spends less computation times and it can converge to the global optimum faster than BRKGA, especially for large instances.

As perspectives, we present short term and long term further works. Con-

cerning the short term works, it may be possible to improve the results obtained from BRKGA and IBMOLS. In BRKGA process, the hybrid decoding step and the elite set selection method can be modified. The two single decodings that were used and combined to be the hybrid decoding, can be changed. Each of these single decodings may be re-defined to mainly consider one objective. For the elite set selection, other indicators can be used like the selection method borrowed from SPEA2. In IBMOLS process, other strategies can be included in the initial population generation step for the first iteration by using useful data of the problem instances. For example, the order to assign each acquisition in the initial sequence can depend on the acquiring duration time of each acquisition. The acquisition with the longest acquiring duration time should be assigned firstly in the sequence. Moreover, the number of removed elements in the perturbation for the initial population generation in the other iterations can also be modified.

For the long term works, more advanced decoding methods can be applied for BRKGA, e.g. consider the decoder as a full multi-objective problem. In IBMOLS process, other perturbation rules and other neighborhood structures may be used. An example of perturbation rule is to insert some feasible acquisitions for replacing the removed elements. For the scheduling problem of an agile satellite, the second objective function, which ensures the fairness of resource sharing, can be modified, e.g. minimizing the sum of profit difference between users. Furthermore, other objective functions can also be included in the problem.

Résumé long

Introduction et contexte

Ce travail s'intéresse à un problème d'optimisation multi-objectif associé à la sélection et l'ordonnancement des observations d'un satellite agile d'observation de la Terre. Nous considérons le cas où de multiples utilisateurs émettent des requêtes pour le satellite. Un algorithme génétique et une recherche locale sont proposés pour résoudre le problème et des expérimentations sont conduites sur des instances réalistes. La mission des satellites d'observation de la Terre (EOS) est d'obtenir des photographies de la surface terrestre, afin de satisfaire les exigences des utilisateurs. Les EOS peuvent acquérir des photographies pendant qu'ils sont en mouvement sur leurs orbites. Il leur faut une période de plusieurs jours pour effectuer une orbite. La surface totale de la terre est observée lorsque les satellites achèvent un cycle complet [47]. Les EOS sont occupés de différents instruments fonction de leurs usages, par exemple des caméras optiques ou des caméras infrarouges. La plupart d'entre eux fonctionnent à basse altitude. Ainsi, quand ils se déplacent au-dessus de la zone visible des photographies requises, les photographies peuvent être prises comme illustré sur la figure 2.1. Ensuite, les satellites vont essayer de transférer les données des images acquises directement à la station sol après leur acquisition, si possible. Sinon, les données sont stockées dans la mémoire bord (disponible en quantité limitée) jusqu'à ce que les satellites soient à portée de la station sol.

Parmi les différents types d'EOS, seuls les satellites dits "agiles" sont considérés dans ce travail. Un EOS agile est équipé d'une seule caméra embarquée, mais l'ensemble du satellite peut tourner autour des trois axes : roulis, tangage et lacet. Le satellite utilise un système de contrôle d'orbite pour pouvoir se déplacer selon ces trois degrés de liberté [59]. Cette propriété permet au satellite de prendre les photos demandées à l'aide d'une seule caméra. Deux exemples de satellites agiles sont Pléiades 1A et 1B, qui ont été dévelop-

pés par le CNES. Tel que présenté, un satellite agile peut se déplacer suivant les trois axes. Ainsi, l'instant de départ pour prendre chaque image n'est pas fixe, mais doit être dans un intervalle de temps donné appelé fenêtre de temps. Pour cette raison, un satellite agile possède un avantage important par rapport à un satellite non agile. D'une part, cela conduit à une meilleure efficacité de l'ensemble du système. D'autre part, le problème de sélection et d'ordonnancement des images candidates est plus difficile à résoudre, du fait que l'espace de recherche à parcourir est plus important [61].

Le processus de gestion de satellite commence lorsque plusieurs utilisateurs fournissent des requêtes vers la station sol, ces demandes ne pouvant pas être attribuées directement à un satellite. La station sol doit gérer les requêtes en sélectionnant et ordonnançant les images candidates, selon certaines limitations du satellite, avant que la séquence obtenue ne soit transmise. Si ces demandes sont acquises par le satellite, elles peuvent générer des profits et des gains. Par conséquent, une façon d'obtenir le plus grand profit est que la station sol essaye d'allouer autant de demandes que possible au satellite. Mais le satellite ne peut généralement pas acquérir toutes les demandes au cours d'une seule révolution. Par exemple, le nombre de demandes peut dépasser la capacité du satellite ou plusieurs zones requises doivent être acquises au cours de la même période.

Chaque demande peut être de deux types : mono ou stéréo. Chaque région est prise une seule fois pour les requêtes mono, alors que pour les demandes stéréo chaque zone doit être acquise deux fois dans la même direction mais à partir d'angles différents. Deux formes possibles de demande, point ou polygone, peuvent être demandées. Le point est une petite zone circulaire d'un rayon de moins de 10 km. Le polygone est une zone polygonale allant de 20 à 100 km. Les deux formes doivent être gérées en transformant les demandes en plusieurs formes rectangulaires appelés bandes. Chaque bande peut être prise en une fois par l'appareil photo du satellite. Un point est considéré comme une seule bande. Si la zone est trop grande pour être prise en une seule fois, un polygone est décomposé en plusieurs bandes. Toutes les bandes ont la même largeur mais des longueurs variables. Un exemple de formes demandées et d'ordre d'acquisition de bandes après gestion est illustré figure 2.2. Il y a deux directions possibles pour acquérir chaque bande. Les deux directions sont parallèles à la longueur de la bande, mais dans des sens opposés, comme le montre la figure 2.3. Parmi ces deux directions, une seule peut être sélectionnée. La bande associée à une direction possible est appelée une acquisition. Ainsi, chaque bande se compose de deux acquisitions possibles. L'intervalle des dates de départ possibles pour la prise de chaque acquisition peut être calculé, en fonction du sens acquis, des temps de début et de fin de visibilité des deux extrémités de la bande, et de la durée d'acquisition de la bande. Dans le problème d'ordonnancement des observations pour les

EOS agiles, certaines acquisitions seront sélectionnées et programmées afin d'obtenir la séquence, qui constitue la solution de ce problème. La séquence obtenue doit satisfaire les contraintes impératives du satellite. Les premières contraintes sont liées aux fenêtres de temps, pendant lesquelles les acquisitions sélectionnées doivent être traitées. Le deuxième groupe de contraintes se réfère à des temps de transition suffisants pour déplacer la caméra du satellite, à partir de l'instant de fin de l'acquisition précédente jusqu'à l'instant de départ de la prochaine. Les troisièmes contraintes obligent l'une des deux directions possibles de chaque bande à être sélectionnée pour l'acquisition. Les quatrièmes contraintes implémentent les contraintes stéréo pour les photos stéréo : si l'une des acquisitions jumelles est sélectionnée, l'autre doit également être sélectionnée, où les acquisitions jumelles prennent la même bande, dans la même direction, mais sous des angles différents.

Chaque acquisition génère un profit. Ainsi, pour le problème d'ordonnancement des observations, l'objectif est la maximisation du profit total. Ce profit total est calculé en fonction de la zone d'acquisition de chaque demande. Le profit de chaque requête d'acquisition peut être calculé en utilisant une fonction linéaire par morceaux du gain. Cette fonction est associée à une fraction de la surface utile acquise sur toute la surface de chaque requête, comme l'illustre la figure 2.4. La requête, dont plus de surface de celle-ci est prise, donne également plus de profit.

Il existe plusieurs études concernant les EOS agiles. Par exemple, une combinaison d'algorithmes génétiques et de recuit simulé a été proposée pour résoudre ce problème dans [62]. La performance de l'algorithme proposé a été comparée à celle d'un recuit simulé seul. Dans [61], quatre méthodes composées d'un algorithme glouton, d'une procédure de programmation dynamique, d'un modèle de programmation par contraintes, et d'une méthode de recherche locale ont été appliquées afin de résoudre une version simplifiée du problème d'ordonnancement pour les EOS agiles.

Le challenge ROADEF 2003 considérait le problème de gestion des missions des EOS agiles (voir `http://challenge.roadef.org/2003/fr/`). Le challenge vise à trouver un ordonnancement réalisable qui maximise le profit total, calculé à partir de la somme des gains des requêtes, qui sont associés à l'acquisition totale ou partielle de chaque requête. La description des données et les critères d'optimisation sont expliqués dans [82]. Le gagnant de ce challenge a utilisé un algorithme basé sur le recuit simulé pour résoudre le problème d'ordonnancement [57]. Le deuxième lauréat a proposé un algorithme basé sur la recherche Tabou [22]. Les auteurs ont adapté l'algorithme de recherche Tabou unifié [21], qui a été développé pour le problème de tournées de véhicules avec fenêtres de temps. En outre, un algorithme de recherche Tabou hybridé avec une recherche systématique a été appliqué pour résoudre ce problème dans [47]. Ces travaux ont considéré le problème

d'ordonnancement pour un EOS agile comme un problème d'optimisation mono-objectif (maximisation du profit total).

Notre travail considère le problème d'ordonnancement des acquisitions d'un EOS agile, où les requêtes émanent de plusieurs utilisateurs. Nous devons optimiser deux fonctions objectif, liées au fait de maximiser un profit total et d'assurer en même temps l'équité du partage des ressources entre tous les utilisateurs. Ainsi, ce problème est modélisé comme un problème d'optimisation multi-objectif. Le deuxième objectif, qui est ajouté afin d'assurer l'équité, revient à minimiser la différence maximale de profits entre utilisateurs. Certains chercheurs ont étudié les problèmes d'optimisation multi-objectif pour les applications spatiales [4][36][84]. En outre, quelques articles considèrent comme objectif l'équité entre les utilisateurs [58]. Des utilisateurs finaux multiples d'EOS agiles ont été considérés et des principes de partage ont été adoptés pour sélectionner le sous-ensemble des candidats en fonction des niveaux d'utilités. Dans [8] et [60], l'utilisation de deux fonctions objectif liées à l'équité et efficacité a été proposée. Trois options ont été discutées pour résoudre ce problème de partage : la première donne la priorité à l'équité, la seconde à l'efficacité, et la troisième calcule un ensemble de compromis pour aider l'humain à prendre des décisions. Pour les méthodes multicritères, au lieu de construire un ensemble complet de solutions non-dominées, les auteurs ont cherché uniquement une décision proche de la ligne avec une pente spécifiée sur le plan de la fonction objectif. Dans [13], les requêtes sélectionnées et ordonnancées pour le cas multi-satellite, multi-orbite et multi-utilisateur ont été étudiées, et une recherche tabou a été utilisée pour résoudre le problème. L'équité a été prise en compte, mais elle n'a pas été considérée en tant que fonction objectif. Les auteurs ont emprunté une moyenne pondérée et ordonnée de [86] pour assurer l'équité des solutions. Les expérimentations ont permis de tester ces algorithmes avec les instances de données fournies par le Centre National d'Etudes Spatiales (CNES).

Notre travail propose deux algorithmes métaheuristiques multi-objectif pour la sélection et l'ordonnancement du sous-ensemble de photographies candidates. Ils s'agit d'un algorithme génétique à clés aléatoires biaisées et une recherche locale multi-objectif basée sur des indicateurs. Les deux fonctions objectif pour ce problème d'ordonnancement sont de maximiser le profit total engendré par les photographies acquises et de minimiser la différence maximale de profits entre les utilisateurs. Le second objectif représente l'équité du partage des ressources entre les utilisateurs. En outre, les solutions obtenues doivent également satisfaire les contraintes physiques de l'EOS agile.

Ce document est organisé comme suit. Le chapitre 3 explique l'algorithme génétique à clés aléatoires biaisées. La recherche locale multi-objectif basée

sur des indicateurs est décrite dans le chapitre 4. Le chapitre 5 présente les résultats d'expériences. Enfin, les conclusions de cette étude et les perspectives de travail sont discutées en fin de document.

Algorithme génétique à clés aléatoires biaisées

Les algorithmes génétiques sont des méthodes de recherche métaheuristiques, qui peuvent résoudre les problèmes de grande taille et obtenir des solutions satisfaisantes en un temps acceptable [77]. Ils commencent par générer une population de p chromosomes, puis appliquent trois mécanismes : la sélection, le croisement et la mutation, pour créer les nouveaux chromosomes pour la prochaine génération et ils itèrent jusqu'à ce que certaines conditions d'arrêt soient rencontrées. Dans ce chapitre, nous utilisons un algorithme génétique à clés aléatoires biaisées (BRKGA) [44], qui combine un algorithme génétique et le concept de clé aléatoire, pour résoudre le problème d'ordonnancement des observations pour un EOS agile. BRKGA a différents moyens pour sélectionner les deux parents de l'opération de croisement, par rapport à l'algorithme génétique à clés aléatoires original (RKGA) [9].

Sélection, croisement et mutation

Avec BRKGA, la population de la nouvelle génération est le résultat de trois étapes : la sélection, le croisement et la mutation, comme illustré figure 1.6 [45].

La première étape est celle de la sélection. Un ensemble d'élite, qui contient pe chromosomes préférés, est choisi dans la population actuelle. Dans ce travail, nous adaptons BRKGA pour résoudre un problème d'optimisation multi-objectif. L'indice de qualité de chaque chromosome doit tenir compte de toutes les fonctions objectif. Nous pouvons choisir une méthode de sélection parmi plusieurs algorithmes efficaces, par exemple, un algorithme génétique de tri non dominé (NSGA-II) [25], un algorithme d'optimisation multi-objectif évolutif de sélection à métrique $\mathscr{S}$ (SMS-EMOA) [12], ou un algorithme évolutif basé sur des indicateurs (IBEA) [92]. Dans ce travail, nous avons opté pour IBEA et son concept d'hyper-volume puisque des expérimentations précédentes montrent qu'il est plus efficace. L'IBEA adapté [92] est appliqué dans ce travail, donc les valeurs objectif sont normalisées dans l'intervalle $[0, 1]$ en utilisant les valeurs limites supérieure et inférieure de chaque objectif. Toutefois, le point de référence ne doit pas être égal à $(0, 1)$, car la valeur de l'hyper-volume aux points extrêmes de l'espace objectif ne peut pas être calculée. Ainsi, pour le problème considéré, qui est de maximiser le premier objectif et de minimiser le deuxième objectif, le point de référence pour calculer l'hyper-volume de chaque solution est pris égal à

$(-1, 2)$, comme illustré sur la figure 3.9. Ensuite, l'indice de qualité de toutes les solutions est calculée. La sélection est réalisée en supprimant la pire des solutions avec la plus faible valeur d'indice de qualité de la population et en mettant à jour les valeurs d'indice de qualité des solutions restantes. La pire solution est éliminée jusqu'à ce que le nombre de solutions restantes satisfasse la taille recommandée de l'ensemble élite pour BRKGA.

La deuxième étape concerne l'ensemble mutant. Il s'agit d'un ensemble de p_m chromosomes générés pour éviter d'être pris au piège dans un optimum local. Ces chromosomes sont générés aléatoirement par la même méthode utilisée pour générer la population initiale.

La dernière partie est la partie de croisement pour lequel chaque descendant de croisement est construit à partir d'un chromosome élite et d'un chromosome de la population actuelle. Chaque élément du descendant de croisement est obtenu à partir d'un élément dans le chromosome élite avec la probabilité ρ_e. Le descendant de croisement est stocké pour remplir l'espace restant de chromosomes dans la population suivante. Par conséquent, la taille de l'ensemble des descendants de croisement est égal à $p - p_e - p_m$. Les valeurs recommandées des paramètres sont affichées dans le tableau 1.1.

Codage et décodage

BRKGA utilise des étapes de codage et de décodage pour la gestion de la solution. Le chromosome à clés aléatoires est généré dans l'étape de codage. Chaque chromosome est formé de plusieurs gènes, qui sont codés par des valeurs réelles générées de façon aléatoire dans l'intervalle $[0, 1]$. Il représente une solution. Pour notre problème, chaque gène représente une acquisition, qui est la bande associée à une direction possible. Ainsi, la taille du chromosome est égale à deux fois le nombre de bandes. Les opérateurs d'algorithmes génétiques sont utilisés pour faire fonctionner ces chromosomes à clés aléatoires pour générer la population de la prochaine génération.

L'autre étape importante de BRKGA est celle de décodage. Elle est utilisée pour transformer le chromosome à clés aléatoires en une solution pour le problème considéré. Pour le problème d'ordonnancement des observations, chaque chromosome est décodé afin d'obtenir une séquence des acquisitions sélectionnées. Dans cette étape de décodage, la priorité de sélection et d'ordonnancement de chaque acquisition est considérée en fonction de la valeur du gène associé. L'acquisition qui a la priorité la plus élevée est considérée en premier pour être ordonnancée dans la séquence. Trois méthodes de décodage sont appliquées pour résoudre ce problème d'ordonnancement des observations : un décodage de base, un décodage de la valeur du gène avec une combinaison de priorité idéal, et un décodage hybride.

Décodage de base

La priorité du décodage de base est définie en utilisant directement la valeur du gène. Cette expression de décodage est donnée par

$$Priority_j = gene_j.$$

Ce décodage de base donne la plus haute priorité pour l'ordonnancement à l'acquisition dont la valeur du gène est maximale. Ainsi, la priorité de cette méthode de décodage ne dépend que de la valeur de la clé aléatoire.

Décodage de la valeur du gène avec une combinaison de priorité idéale

Cette méthode de décodage est empruntée à [65]. Elle a été appliquée pour le problème d'ordonnancement de projet sous contraintes de ressources (RCPSP). Ce décodage définit la priorité de chaque acquisition en fonction de deux valeurs : la valeur du gène associé comme dans le décodage de base et sa valeur de priorité idéale calculée. Pour le concept de priorité idéale dans [65], le travail, qui a l'instant de départ possible le plus tôt, doit être choisi en premier et être ordonnancé en début de la séquence de solutions. Par conséquent, la priorité idéale donne une plus grande priorité pour sélectionner et ordonnancer le travail avec une date de départ possible au plus tôt. Cette priorité idéale est donnée par

$$\frac{LLP_j}{LCP},$$

où LLP_j est la longueur du chemin le plus long depuis le début du travail j jusqu'à la fin du projet et LCP est la longueur du chemin critique du projet.

Le coefficient qui ajuste la priorité pour tenir compte des valeurs des gènes du chromosome à clés aléatoires est donnée par $(1 + gene_j)/2$. Ainsi, l'expression de décodage de chaque travail j est

$$Priority_j = \frac{LLP_j}{LCP} \times \left[\frac{1 + gene_j}{2}\right]$$

Dans [65], la minimisation de la durée totale d'ordonnancement (makespan) est le seul critère pris en compte. Dans notre travail, la notion de priorité idéale est modifiée afin d'être plus appropriée pour résoudre le problème d'ordonnancement des observations multi-objectif. Par conséquent, la priorité idéale donne une plus grande priorité pour sélectionner et ordonnancer l'acquisition avec un temps de départ possible au plus tôt. Cette priorité idéale est donnée par

$$\frac{Tmax_L - Tmin_j}{Tmax_L},$$

où $Tmax_L$ est la date de début le plus tard de la dernière acquisition possible et $Tmin_j$ la date de début au plus tôt de l'acquisition j.

Le même facteur que dans [65] est utilisé pour régler la priorité. Ainsi, l'expression de décodage de la valeur du gène combinée à la priorité idéale pour chaque acquisition j est

$$Priority_j = \frac{Tmax_L - Tmin_j}{Tmax_L} \times \left[\frac{1 + gene_j}{2}\right].$$

Un exemple de calcul de priorité idéale de la deuxième méthode de décodage est illustré figure 3.4. Il est appliqué au problème d'ordonnancement d'observation d'un EOS, qui doit sélectionner et ordonnancer quatre acquisitions, a, b, c et d. Pour cet exemple, l'ordre des acquisitions, qui sera considéré comme étant affecté à la séquence selon la priorité idéale, est le suivant: b, c, d, a.

La priorité idéale, qui est utilisée dans l'équation de ce second décodage, est calculée en utilisant les valeurs des données du problème. Il peut rendre cette méthode de décodage plus rapide pour atteindre une solution optimale, qui peut être un optimum local. Par conséquent, il faut être prudent avec la prise au piège dans un optimum local.

Décodage hybride

Enfin, nous proposons la troisième méthode de décodage. Elle combine les première et deuxième méthodes de décodage. Cette méthode hybride obtient deux solutions d'un chromosome. Lors de l'application du décodage hybride, les méthodes pour gérer l'ensemble d'élite doivent être définies. Plusieurs méthodes différentes peuvent être utilisées pour sélectionner l'ensemble d'élite. Toutefois, une méthode de gestion de l'ensemble élite est présentée dans ce travail. Chaque chromosome de la population est décodé à partir des deux méthodes de décodage. La première solution est obtenue à partir du décodage de base et la seconde solution est obtenue à partir du décodage de la valeur du gène avec combinaison de priorité idéale. Ensuite, les deux solutions, avec la valeur de la fonction objectif correspondante, sont comparées à l'aide de la relation de dominance au sens de Pareto. Si une solution peut dominer l'autre, la solution dominante est choisie pour être stockée dans l'ensemble des solutions. Dans le cas contraire, l'une des deux solutions est choisie au hasard. Le processus de décodage est répété jusqu'à ce que tous les chromosomes de la population soient décodés. Quand il termine, la taille de l'ensemble des solutions est égale à p. Ensuite, les p_e solutions sont sélectionnées pour devenir l'ensemble élite en utilisant les mêmes méthodes avec un seul décodage. Le principe de la gestion de l'ensemble élite est représenté sur la figure 3.10.

Au cours de l'étape de décodage, les contraintes impératives sont vérifiées pour chaque acquisition séquentiellement en fonction de sa priorité. Chaque acquisition considérée peut être affectée à la séquence, seulement si la séquence obtenue peut satisfaire toutes les contraintes. L'organigramme de vérification de contraintes et d'affectation d'acquisitions est indiqué sur la figure 3.5. L'exemple d'une solution de l'instance de plus petite taille est représenté sur la figure 3.6. Cette instance se compose de deux bandes. Ainsi, le nombre de gènes à clés aléatoires, qui sont associés aux acquisitions, est égal à quatre. Cet exemple montre la solution décodée à partir du décodage de base (la priorité pour sélectionner et ordonnancer chaque acquisition est égale à sa valeur de gène). L'étape de décodage est utilisée pour obtenir la séquence d'acquisitions sélectionnées et les valeurs des deux fonctions objectif.

Recherche locale multi-objectif basée sur des indicateurs

Dans ce chapitre, nous proposons une approche de recherche locale multi-objectif basée sur des indicateurs (IBMOLS) pour résoudre le problème d'ordonnancement des observations multi-utilisateurs pour un EOS agile. IBMOLS est un algorithme générique, qui combine l'utilisation de la recherche locale de base et d'un indicateur binaire de l'algorithme évolutionnaire basé sur des indicateurs. Il a été proposé initialement dans [7].

Adapté à notre problème bi-objectif, IBMOLS fonctionne comme suit. Le front de Pareto approximé PO est initialisé à l'ensemble vide et il est mis à jour à la fin de chaque itération. Deux procédures sont utilisées dans ce travail. La première est appliquée pour la première itération et la seconde est appliquée pour les autres itérations. Ensuite, les solutions non dominées de la population sont stockées dans l'ensemble archive A. Les valeurs d'indice de qualité de tous les individus de la population sont calculées en utilisant l'indicateur basé sur le concept d'hyper-volume de [92] et l'étape de recherche locale est appliquée pour chaque individu. Après cela, la population mise à jour est combinée avec l'ensemble archive A et les solutions non dominées de cet ensemble combiné sont stockées dans le nouvel ensemble archive A. Si l'ensemble archive A change, le processus applique à nouveau l'étape de recherche locale. Sinon, cette itération est terminée et l'ensemble archive A final est obtenu. Ensuite, le front de Pareto approximé PO est mis à jour en combinant l'ensemble archive obtenu A avec le front de Pareto approximé PO, et l'ensemble des solutions non dominées de l'ensemble combiné devient alors le nouveau front de Pareto PO. L'itération suivant se poursuit, si l'algorithme ne satisfait pas les critères d'arrêt.

Génération de population – première itération

Pour la première itération d'IBMOLS, N individus sont générés au hasard pour composer la population initiale. Chaque individu représente une solution, qui est une séquence des acquisitions sélectionnées.

L'organigramme de la génération de la population initiale pour la première itération est représenté figure 4.2. Toutes les acquisitions sont assignées pour être les éléments de l'ensemble d'acquisitions sélectionnées. Pour chaque acquisition selon un ordre aléatoire, il est vérifié qu'elle satisfait à la contrainte de temps de transition suffisant et la contrainte de fenêtre de temps. Si elle satisfait les deux contraintes, la date de départ est calculée et fixée dans l'ensemble des dates de départ. Par ailleurs, l'acquisition, qui concerne la direction acquise opposée de la même bande, est retirée de l'ensemble des acquisitions sélectionnées. Sinon, l'acquisition considérée est retirée. Le procédé pour la vérification de ces contraintes est répété jusqu'à ce que toutes les acquisitions de l'ensemble d'acquisitions sélectionnées soient testées. Après cela, les ensembles temporaires des acquisitions sélectionnées et des dates de départ sont obtenus. Dans cette étape, la contrainte stéréo doit être vérifiée pour chaque acquisition sélectionnée une par une dans l'ensemble temporaire. Si l'acquisition considérée vient d'une bande stéréo, sa jumelle doit également être assignée. Si cette acquisition liée n'est pas assignée, l'acquisition considérée est retirée. Lorsque toutes les acquisitions sélectionnées sont vérifiées, l'ensemble des dates de départ est recalculé.

Génération de population – autres itérations

Dans IBMOLS, une recherche locale itérative est utilisée pour rechercher les solutions non-dominées en commençant la recherche à partir de différentes populations initiales. Un mécanisme de perturbation est appliqué afin d'échapper à des optima locaux. Toutefois, le nombre de composants de solution modifiés doit être défini avec précision. Si la perturbation est trop forte, de meilleures solutions peuvent être trouvées, mais avec une très faible probabilité. D'autre part, si la perturbation est trop faible, la recherche locale va retomber dans l'optimum local qui vient juste d'être visiter [63].

Pour la perturbation, un individu est généré en modifiant une solution du front de Pareto approximé PO de l'itération courante. Dans ce travail, les solutions du front de Pareto approximé sont choisies aléatoirement. Le nombre de solutions sélectionnées est égal à la taille N de la population initiale. Chaque solution contient l'ensemble d'acquisitions sélectionnées. Il est modifié en supprimant certaines acquisitions à la position aléatoire j de l'ensemble d'acquisitions sélectionnées. Le nombre d'éléments éliminé est d'environ $1/4$ de la taille n_{ori} de l'ensemble d'acquisitions sélectionnées initial. Par ailleurs, au cours de l'élimination, la contrainte stéréo doit être vérifiée. Si

l'acquisition retirée fait partie d'une demande stéréo, sa jumelle doit également être retirée. L'élimination d'acquisition est répétée jusqu'à ce que le nombre d'acquisitions restantes dans l'ensemble d'acquisitions sélectionnées ou n_{modi} soient inférieur ou égal à $3/4$ de la taille n_{ori} de l'ensemble original. Ensuite, l'ensemble d'acquisitions sélectionnées modifié deviendra la partie de l'individu, qui est un membre de la population initiale dans l'itération suivante. La génération de la population à l'aide de la perturbation est expliquée dans l'algorithme 7.

Dans le processus de perturbation, nous devons éviter la génération d'une solution déjà visitée. Par conséquent, le nombre d'acquisitions enlevées est pré-calculé. S'il est inférieur ou égal à un, la perturbation va générer l'individu, qui a été visité. Dans ce cas, la génération aléatoire sera utilisée pour générer l'individu, au lieu d'utiliser la perturbation.

Etape de recherche locale

L'étape de recherche locale commence à partir d'un individu de la population P et se déplace itérativement vers un voisin. Dans l'IBMOLS original, une première stratégie d'amélioration est utilisée pour sélectionner le voisin. Cependant, dans ce travail, une meilleure stratégie d'amélioration est préférable. Au cours de l'exploration du voisinage, les valeurs d'indice de qualité de chaque voisin sont calculées. Le voisin avec la meilleure indice de qualité est généré et sélectionné pour remplacer la plus mauvaise solution dans la population. Le voisinage de tous les individus de la population P est exploré. La population et l'ensemble archive sont mis à jour. Si l'ensemble archive mis à jour A ne change pas, l'étape de recherche locale est arrêtée. Sinon, une autre étape de la recherche locale est effectuée.

Le voisinage qui est utilisé dans ce travail se compose de quatre types de mouvement: i) insérer une acquisition mono i ; ii) retirer une acquisition mono i ; iii) insérer des acquisitions jumelles i et j, et iv) supprimer des acquisitions jumelles i et j. Pour chaque type de mouvement, toutes les acquisitions mono ou toutes les paires d'acquisitions jumelles sont essayées pour être insérées ou retirées une par une.

Parmi les quatre types de mouvement, dans le cas de l'insertion, la faisabilité du voisin doit être vérifiée. Une acquisition non-assignée peut être insérée à une position de l'ensemble des acquisitions sélectionnées uniquement si toutes les contraintes impératives sont satisfaites. Pour l'acquisition mono, les trois contraintes, qui consistent en la contrainte de fenêtre de temps, la contrainte de temps de transition suffisant et la contrainte que la même bande ne peut être sélectionnée et acquise que dans une seule direction, doivent être vérifiées. De plus, pour l'acquisition stéréo, la contrainte stéréo doit être vérifiée.

On calcule la plus grande date de début au plus tard des acquisitions sélectionnées, qui sont ordonnancées derrière la position d'insertion, avant de vérifier la faisabilité. Elles sont ordonnancées pour être prises le plus tard possible c'est-à-dire à leur date de début au plus tard. Ainsi, le plus grand espace est libéré pour supporter l'insertion de l'acquisition non-assignée. Par exemple, l'acquisition non-assignée $Acq\ k$ est choisie pour tenter d'être insérée devant l'acquisition sélectionnée sa_3. Les autres acquisitions suivant sa_3 peuvent être déplacées vers la droite le plus tard possible. Cet exemple est illustré sur la figure 4.7.

Résultats expérimentaux

Les méthodes ont été testées sur des instances modifiées du challenge ROADEF 2003 (Testset A). Elles sont modifiées pour les exigences de 4 utilisateurs et le format des noms d'instance est modifié pour devenir `a_b_c`, où `a` est le nombre de demandes, `b` est le nombre de demandes stéréo, et `c` est le nombre de bandes.

Pour l'algorithme génétique à clés aléatoires biaisées (BRKGA), les valeurs des paramètres de l'algorithme ont été expérimentalement réglées pour notre travail. La taille de la population de BRKGA est fixée égale à la longueur du chromosome à clés aléatoires ou à deux fois le nombre de bandes. Les tailles des trois parties, qui sont composées pour être la population pour la prochaine génération, sont réglées en accord avec les valeurs recommandées du tableau 1.1. La taille de l'ensemble élite est égal au nombre d'ordonnancements non-répétés des solutions non-dominées, mais elle n'est pas plus grande que $0.15p$. La taille de l'ensemble mutant est égale à $0.3p$. La probabilité d'héritage d'élément élite pour l'opération de croisement est réglée à 0.6. A chaque itération, les solutions non-dominées sont stockées dans une archive. S'il existe au moins une solution de la population actuelle qui peut dominer des solutions dans l'archive, l'archive sera mise à jour. Par conséquent, nous utilisons le nombre d'itérations depuis la dernière amélioration de l'archive pour être un critère d'arrêt. La valeur d'arrêt est définie à 50. Trois méthodes de décodage sont considérées dans ce travail. Pour la recherche locale multi-objectif basée sur indicateur (IBMOLS), les résultats obtenus sont comparés avec les résultats de BRKGA. Les valeurs des paramètres de taille de la population et les valeurs d'arrêt doivent être réglées par les expérimentations. Nous utilisons le nombre d'itérations depuis la dernière amélioration du front de Pareto approximé comme critère d'arrêt. La taille des populations de 10 et la valeur d'arrêt de 50 itérations sont utilisées.

Les deux algorithmes proposés sont implémentés en C++ et trente exécutions par instance sont testées. Les hyper-volumes du front de Pareto ap-

proximé sont calculés en utilisant un point de référence de 0 pour le premier objectif (maximiser le profit total) et le maximum des sommations de profit de chaque utilisateur pour le second (minimiser la différence de profit entre les utilisateurs). Les valeurs des hyper-volumes, qui sont obtenues à partir des deux algorithmes proposés, sont tracées par des diagrammes en boîte.

Dans le premier type de comparaison, on oppose principalement les résultats des différentes méthodes de décodage de BRKGA, qui sont le décodage de base (D1), le décodage de la valeur de gène avec la combinaison de priorité idéale (D2), et le décodage hybride (HD). Les diagrammes en boîte des valeurs des hyper-volumes et les temps de calcul moyens sont présentés dans la figure 5.7.

Pour la plus petite instance (instance 2_0_2), les résultats ne peuvent pas être atteints. En effet, la taille de la population, qui est égale au double du nombre de bandes, est trop petite pour générer la nouvelle génération à partir des trois ensembles de chromosomes dans le processus BRKGA. La plupart des résultats montrent que le décodage hybride obtient les meilleurs résultats, comparé aux deux décodages simples. En effet, le décodage hybride cumule les avantages des deux décodages simples pour toutes les instances. Par exemple, dans l'instance 12_2_25, la première méthode de décodage obtient de meilleurs résultats que la seconde, ainsi le décodage hybride obtient des résultats similaires à la première méthode. Pour l'instance 77_40_147, le décodage hybride obtient des résultats similaires au second décodage, qui obtient de meilleurs résultats que le premier. Ainsi, la méthode de décodage hybride est efficace pour résoudre la plupart des instances. Par rapport à D1, il peut réduire la gamme des valeurs des hyper-volumes. Cela signifie que le décodage hybride peut fournir des résultats avec de meilleurs écarts-types. En outre, pour certaines instances où le second décodage se confine dans un optimum local, le décodage hybride est capable d'atteindre l'optimum global. En ce qui concerne le temps de calcul, la méthode de décodage hybride passe plus de temps à chaque itération, mais elle peut obtenir de bonnes solutions en un temps de calcul global raisonnable.

Dans le deuxième type de comparaison, nous comparons principalement les résultats entre BRKGA et IBMOLS. Pour BRKGA, les résultats sont obtenus à partir du décodage hybride, qui utilise la méthode de sélection de l'algorithme évolutionnaire basé sur des indicateurs (IBEA) pour sélectionner les chromosomes préférés pour devenir l'ensemble élite. En outre, la sélection de la solution dominante est utilisée pour gérer l'ensemble élite dans le processus de décodage hybride. Pour IBMOLS, les résultats sont obtenus en utilisant la structure de voisinage, qui consiste en l'insertion et le retrait des acquisitions mono et stéréo. La méthode de vérification de la faisabilité, qui calcule la date de début au plus tard avant de vérifier la faisabilité de l'insertion, est appliquée. Pour générer la population initiale, la génération aléatoire est

utilisée dans la première itération et la perturbation est appliquée dans les autres itérations. Le nombre de 50 itérations depuis la dernière amélioration d'archive est utilisé comme critère d'arrêt pour les deux méthodes, BRKGA et IBMOLS. Les diagrammes en boîte des valeurs des hyper-volumes et le temps de calcul moyen de BRKGA et IBMOLS sont présentés sur la figure 5.13. La première colonne illustre les résultats de BRKGA, la deuxième colonne ceux d'IBMOLS.

Les diagrammes en boîte montrent qu'IBMOLS obtient des valeurs médianes des hyper-volumes meilleures pour toutes les instances, de même que de meilleurs écarts-types pour la plupart des résultats. En outre, IBMOLS utilise moins de temps de calcul que BRKGA, en particulier pour les grandes instances. La figure 5.14 représente l'amélioration des valeurs des hyper-volumes en fonction du temps de calcul pour les instances moyennes et grandes. Dans chaque graphique, l'amélioration des valeurs des hyper-volumes entre le processus BRKGA et le processus IBMOLS est analysé. Les résultats montrent qu'IBMOLS obtient des solutions plus proches de l'optimum global pour les instances moyennes et grandes. En outre, IBMOLS peut converger vers l'optimum global plus rapidement que BRKGA.

Enfin, les meilleurs fronts de Pareto approximés de toutes les instances sont illustrés sur la figure 5.15. Pour chaque instance, le profit total est présenté en abscisse et la différence maximale de profits entre les utilisateurs est présentée en ordonnée.

Conclusions

Un algorithme génétique à clés aléatoires biaisées (BRKGA) et une recherche locale multi-objectif basée sur des indicateurs (IBMOLS) sont utilisés pour résoudre un problème d'optimisation multi-objectif associé à la sélection et l'ordonnancement des prises de vue d'un satellite agile d'observation de la Terre. Les instances du challenge ROADEF 2003 sont modifiées afin de prendre en compte explicitement les exigences de 4 utilisateurs. Deux fonctions objectif, maximiser le profit total et minimiser la différence maximale de profits entre les utilisateurs pour l'équité du partage des ressources, sont considérées et les contraintes impératives doivent être respectées.

Pour BRKGA, le codage à clés aléatoires génère chaque chromosome de la population et les chromosomes sont décodés pour être les séquences des acquisitions sélectionnées. Trois méthodes de décodage, un décodage de base, un décodage de la valeur du gène avec combinaison de priorité idéale, et un décodage hybride, sont présentés dans ce document. La méthode de sélection des élites d'IBEA est utilisée pour sélectionner les solutions préférées pour devenir l'ensemble élite de la population. Un ensemble élite, un ensemble des

descendants du croisement, et un ensemble mutant sont combinés pour former la population suivante. Les valeurs des hyper-volumes des trois méthodes de décodage sont comparées. La plupart des résultats montrent que le décodage hybride obtient de meilleures valeurs moyennes des hyper-volumes, de même qu'il peut réduire la gamme des valeurs des hyper-volumes en des temps de calcul raisonnables.

En outre, IBMOLS est implémenté par la génération de la population initiale en utilisant une génération aléatoire pour la première itération et une perturbation pour les autres itérations. L'affectation de la valeur de la fonction d'évaluation basée sur des indicateurs avec le concept d'hyper-volume d'IBEA est appliquée pour comparer les solutions dans la population. Les valeurs des hyper-volumes d'IBMOLS et de BRKGA sont comparées. La plupart des résultats montrent qu'IBMOLS obtient de meilleures solutions et nécessite moins de temps de calcul.

Bibliography

[1] Ravindra K. Ahuja, Özlem Ergun, James B. Orlin, and Abraham P. Punnen. A survey of very large-scale neighborhood search techniques. *Discrete Appl. Math.*, 123(1-3):75–102, November 2002.

[2] I. Alaya, C. Solnon, and K. Ghedira. Ant colony optimization for multi-objective optimization problems. In *19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), 2007*, volume 1, pages 450 –457, oct. 2007.

[3] J. Andersson. A survey of multiobjective optimization in engineering design. Technical report, Department of Mechanical Engineering, Linköping University, Linköping, Sweden, 2000.

[4] A. Arias-Montaño, C.A.C. Coello, and E. Mezura-Montes. Multiobjective evolutionary algorithms in aeronautical and aerospace engineering. *IEEE Transactions on Evolutionary Computation*, 16(5):662 –694, oct. 2012.

[5] S. Bandyopadhyay, S. Saha, U. Maulik, and K. Deb. A simulated annealing-based multiobjective optimization algorithm: Amosa. *IEEE Transactions on Evolutionary Computation*, 12(3):269 –283, june 2008.

[6] Alec Banks, Jonathan Vincent, and Chukwudi Anyakoha. A review of particle swarm optimization. part II: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications. 7(1):109–124, March 2008.

[7] M. Basseur and E.K. Burke. Indicator-based multi-objective local search. In *IEEE Congress on Evolutionary Computation (CEC), 2007*, pages 3100 –3107, sept. 2007.

[8] N. Bataille, M. Lemaître, and G. Verfaillie. Efficiency and fairness when sharing the use of a satellite. In *Artificial Intelligence, Robotics and Automation in Space, Proceedings of the Fifth International Symposium, ISAIRAS '99*, pages 465 –470, June 1999.

[9] James C. Bean. Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal on Computing*, 6(2):154–160, 1994.

[10] E. Bensana, M. Lemaître, and G. Verfaillie. Earth observation satellite management. *Constraints*, 4:293–299, 1999.

[11] D. Bertsimas and J. Tsitsiklis. Simulated annealing. *Statistical Science*, 8(1):10–15, 1993.

[12] Nicola Beume, Boris Naujoks, and Michael Emmerich. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669, 2007.

[13] Nicola Bianchessi, Jean-François Cordeau, Jacques Desrosiers, Gilbert Laporte, and Vincent Raymond. A heuristic for the multi-satellite, multi-orbit and multi-user management of earth observation satellites. *European Journal of Operational Research*, 177(2):750–762, 2007.

[14] Nicola Bianchessi and Giovanni Righini. Planning and scheduling algorithms for the cosmo-skymed constellation. *Aerospace Science and Technology*, 12(7):535 – 544, 2008.

[15] Christian Blum and Andrea Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, 35(3):268–308, September 2003.

[16] S. Bouveret. Comparison of two constraint programming algorithms for computing leximin-optimal allocations. In *Proceedings of the Workshop on Modelling and Solving Problems with Constraints*, Riva del Garda, Italy, August 2006.

[17] Edwin K.P. Chong and Stanislaw H. Żak. *An Introduction to Optimization*. Wiley-Interscience, 2001.

[18] C.A.C. Coello, G.T. Pulido, and M.S. Lechuga. Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3):256 – 279, june 2004.

[19] Carlos A. Coello Coello. A short tutorial on evolutionary multiobjective optimization. In *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, EMO '01, pages 21–40, London, UK, UK, 2001. Springer-Verlag.

[20] Y. Collette and P. Siarry. *Multiobjective Optimization. Principles and Case Studies*. Springer, 2003.

[21] J.-F. Cordeau, G. Laporte, and A. Mercier. A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational Research Society*, 52:928–936, 2001.

[22] Jean-François Cordeau and Gilbert Laporte. Maximizing the value of an earth observation satellite orbit. *Journal of the Operational Research Society*, 56:962–968, 2005.

[23] Y. Crama, A.W.J. Kolen, and E.J. Pesch. Local search in combinatorial optimization. In P.J. Braspenning, F. Thuijsman, and A.J.M.M. Weijters, editors, *Artificial Neural Networks*, volume 931 of *Lecture Notes in Computer Science*, pages 157–174. Springer Berlin Heidelberg, 1995.

[24] Olivier L. De Weck. Multiobjective optimization: History and promise. In *Proc. 3rd China-Japan-Korea Joint Symp. Optimization Structural Mech. Syst. Invited Keynote Paper GL2-2*, Kanazawa, Japan, Oct.–Nov. 2004.

[25] Kalyanmoy Deb, Amrit Pratep, Sameer Agarwal, and T. Meyarivan. A fast and elite multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[26] M. Dorigo. *Optimization, learning and natural algorithms*. PhD thesis, Politecnico di Milano, Italy, 1992.

[27] M. Dorigo and L.M. Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53 –66, apr 1997.

[28] M. Dorigo, V. Maniezzo, and A. Colorni. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 26(1):29–41, feb 1996.

[29] Marco Dorigo and Thomas Stützle. The ant colony optimization metaheuristic: Algorithms, applications, and advances. In *Handbook of Metaheuristics*, pages 251–285. Kluwer Academic Publishers, 2002.

[30] Marco Dorigo and Thomas Stützle. Ant colony optimization: Overview and recent advances. In Michel Gendreau and Jean-Yves Potvin, editors, *Handbook of Metaheuristics*, volume 146 of *International Series in Operations Research & Management Science*, pages 227–263. Springer US, 2010.

[31] Yuval Filmus and Justin Ward. The power of local search: Maximum coverage over a matroid. In *STACS*, pages 601–612, 2012.

[32] R. Fletcher and M. J. D. Powell. A rapidly convergent descent method for minimization. *The Computer Journal*, 6(2):163–168, 1963.

[33] Carlos M. Fonseca and Peter J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In S. Forrest, editor, *Genetic Algorithms: Proceedings of the Fifth International Conference*, pages 416–423, San Mateo, CA, July 1993. Morgan Kaufmann.

[34] Carlos M. Fonseca and Peter J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16, March 1995.

[35] V. Gabrel and C. Murat. Mathematical programming for earth observation satellite mission planning. In T. A. Ciriani, G. Fasano, S. Gliozzi, and R. Tadei, editors, *Operations Research in Space and Air*, pages 103–122. Kluwer Academics, 2003.

[36] Virginie Gabrel and Daniel Vanderpooten. Enumeration and interactive selection of efficient paths in a multiple criteria graph for scheduling an earth observing satellite. *European Journal of Operational Research*, 139(3):533 – 542, 2002.

[37] X. Gandibleux, M. Sevaux, and V. Sörensen, K.; T'Kindt. *Metaheuristics for Multiobjective Optimisation*. Springer, 2004.

[38] Luca Gaspero and Andrea Schaerf. Neighborhood portfolio approach for local search applied to timetabling problems. *Journal of Mathematical Modelling and Algorithms*, 5:65–89, 2006.

[39] Zong Woo Geem, Joong Hoon Kim, and G.V. Loganathan. A new heuristic optimization algorithm: Harmony search. *SIMULATION*, 76(2):60–68, 2001.

[40] A. Globus, J. Crawford, J. Lohn, and A. Pryor. A comparison of techniques for scheduling earth observing satellites. In *Proc. of the 16th Conference on the Innovative Applications of Artificial Intelligence*, 2004.

[41] Al Globus, James Crawford, Jason Lohn, and Anna Pryor. A comparison of techniques for scheduling fleets of earth-observing satellites, 2008.

[42] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, 1997.

[43] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Co., Inc., 1989.

[44] José Fernando Gonçalves and JorgeRaimundo de Almeida. A hybrid genetic algorithm for assembly line balancing. *Journal of Heuristics*, 8:629–642, 2002.

[45] José Fernando Gonçalves and Mauricio G.C. Resende. Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17(5):487–525, 2011.

[46] N. Goulart, S.R. de Souza, L.G.S. Dias, and T.F. Noronha. Biased random-key genetic algorithm for fiber installation in optical network optimization. In *IEEE Congress on Evolutionary Computation (CEC), 2011*, pages 2267–2271, june 2011.

[47] Djamal Habet, Michel Vasquez, and Yannick Vimont. Bounding the optimum for the problem of scheduling the photographs of an agile earth observing satellite. *Computational Optimization and Applications*, 47:307–333, 2010.

[48] D. Henderson, S.H. Jacobson, and A.W. Johnson. The theory and practice of simulated annealing. In F. Glover and G. Kochenberger, editors, *Handbook on Metaheuristics*, pages 287–319. 2003.

[49] Alain Hertz and Marino Widmer. Guidelines for the use of metaheuristics in combinatorial optimization. *European Journal of Operational Research*, 151(2):247–252, 2003.

[50] Xiaohui Hu and R. Eberhart. Multiobjective optimization using dynamic neighborhood particle swarm optimization. In *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, volume 2, pages 1677 –1681, 2002.

[51] D. S. Johnson and L. A.l McGeoch. The traveling salesman problem: a case study in local optimization. In E. H. L. Aarts and J. K. Lenstra, editors, *Local Search in Combinatorial Optimization*. Wiley, New York, 1997.

[52] Li Jufang, Yao Feng, Bai Baocun, and He Renjie. A decomposition-based algorithm for imaging satellites scheduling problem. In *Information Engineering and Computer Science, 2009. ICIECS 2009. International Conference on*, pages 1 –6, dec. 2009.

[53] J. Kennedy and R. Eberhart. Particle swarm optimization. In *IEEE International Conference on Neural Networks, 1995*, volume 4, pages 1942–1948, nov/dec 1995.

[54] J. Kiefer. Sequential minimax search for a maximum. In *Proceedings of the American Mathematical Society*, volume 4, pages 502–506, 1953.

[55] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[56] Abdullah Konak, David W. Coit, and Alice E. Smith. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering & System Safety*, 91(9):992 – 1007, 2006.

[57] Eelco J. Kuipers. An algorithm for selecting and timetabling requests for an earth observation satellite. *Bulletin de la Société Française de Recherche Opérationnelle et d'Aide à la Décision*, Editor Automme-Hiver(11):7–10, 2003.

[58] M. Lemaître, G. Verfaillie, H. Fargier, J. Lang, N. Bataille, and J.M. Lachiver. Sharing the use of earth observation satellites. In *3rd NASA Workshop on Planning and Scheduling*, 2002.

[59] M. Lemaître, G. Verfaillie, and F. Jouhaud. How to manage the new generation of agile earth observation satellites. In *6th International SpaceOps Symposium (Space Operations)*, 2000.

[60] Michel Lemaître, Gérard Verfaillie, and Nicolas Bataille. Exploiting a common property resource under a fairness constraint: a case study. In *Proceedings of the 16th international joint conference on Artifical intelligence - Volume 1*, IJCAI'99, pages 206–211, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[61] Michel Lemaître, Gérard Verfaillie, Frank Jouhaud, Jean-Michel Lachiver, and Nicolas Bataille. Selecting and scheduling observations of agile satellites. *Aerospace Science and Technology*, 6(5):367 – 381, 2002.

[62] Yuqing Li, Minqiang Xu, and Rixin Wang. Scheduling observations of agile satellites with combined genetic algorithm. In *Natural Computation, 2007. ICNC 2007. Third International Conference on*, volume 3, pages 29 –33, aug. 2007.

[63] Helena R. Lourenço, Olivier C. Martin, and Thomas Stützle. Iterated local search. In F. Glover and G. Kochenberger, editors, *Handbook on Metaheuristics*, pages 321–353. 2002.

[64] Mohamed A.A. Mansour and Maged M. Dessouky. A genetic algorithm approach for solving the daily photograph selection problem of the spot5 satellite. *Computers & Industrial Engineering*, 58(3):509 – 520, 2010.

[65] 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. *Computers and Operations Research*, 36(1):92 – 109, 2009.

[66] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 1965.

[67] P. Ngatchou, Anahita Zarei, and M.A. El-Sharkawi. Pareto multi objective optimization. In *Intelligent Systems Application to Power Systems, 2005. Proceedings of the 13th International Conference on*, pages 84 –91, nov. 2005.

[68] Chandrasekharan Rajendran and Hans Ziegler. Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs. *European Journal of Operational Research*, 155(2):426 – 438, 2004.

[69] Glaydston Mattos Ribeiro, Miguel Fragoso Constantino, and Luiz Antonio Lorena. Strong formulation for the spot 5 daily photograph scheduling problem. *J. Comb. Optim.*, 20(4):385–398, November 2010.

[70] M.W.P. Savelsbergh. Local search in routing problems with time windows. *Annals of Operations Research*, 4:285–305, 1985.

[71] J. David Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pages 93–100, Hillsdale, NJ, USA, 1985. L. Erlbaum Associates Inc.

[72] S. N. Sivanandam and S. N. Deepa. *Introduction to Genetic Algorithms*. Springer Publishing Company, Incorporated, 1st edition, 2007.

[73] Krzysztof Socha and Marco Dorigo. Ant colony optimization for continuous domains. *European Journal of Operational Research*, 185(3):1155 – 1173, 2008.

[74] Mei-Ping Song and Guo chang Gu. Research on particle swarm optimization: a review. In *Machine Learning and Cybernetics, 2004. Proceedings of 2004 International Conference on*, volume 4, pages 2236 – 2241 vol.4, aug. 2004.

[75] N. Srinivas and Kalyanmoy Deb. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, September 1994.

[76] A. Suppapitnarm, K. A. Seffen, G. T. Parks, and P. J. Clarkson. A simulated annealing algorithm for multiobjective optimization. *Engineering Optimization*, 33(1):59–85, 2000.

[77] El-Ghazali Talbi. *Metaheuristics: from design to implementation*. John Wiley & Sons, Inc., 2009.

[78] H. Tamaki, H. Kita, and S. Kobayashi. Multi-objective optimization by genetic algorithms: a review. In *IEEE International Conference on Evolutionary Computation, 1996*, pages 517 –522, may 1996.

[79] K.C. Tan, T.H. Lee, and E.F. Khor. Evolutionary algorithms for multi-objective optimization: performance assessments and comparisons. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, volume 2, pages 979 –986 vol. 2, 2001.

[80] Michel Vasquez and Jin-Kao Hao. A "logic-constrained"knapsack formulation and a tabu algorithm for the daily photograph scheduling of an earth observation satellite. *Comput. Optim. Appl.*, 20(2):137–157, November 2001.

[81] Michel Vasquez and Jin-Kao Hao. Upper bounds for the spot 5 daily photograph scheduling problem. *Journal of Combinatorial Optimization*, 7:87–103, 2003.

[82] Gérard Verfaillie, Michel Lemaître, Nicolas Bataille, and Jean-Michel Lachiver. Management of the mission of earth observation satellites challenge description. Technical report, Centre National d'Etudes Spatiales, France, 2002.

[83] Stefan Voß. Meta-heuristics: The state of the art. In Alexander Nareyek, editor, *Local Search for Planning and Scheduling*, volume 2148 of *Lecture Notes in Computer Science*, pages 1–23. Springer Berlin Heidelberg, 2001.

[84] Jun Wang, Ning Jing, Jun Li, and Zhong Hui Chen. A multi-objective imaging scheduling approach for earth observing satellites. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, GECCO '07, pages 2211–2218, New York, NY, USA, 2007. ACM.

[85] Pei Wang and Gerhard Reinelt. A heuristic for an earth observing satellite constellation scheduling problem with download considerations. *Electronic Notes in Discrete Mathematics*, 36:711 – 718, 2010.

[86] R.R. Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transactions on Systems, Man and Cybernetics*, 18(1):183–190, 1988.

[87] Tjalling J. Ypma. Historical development of the newton-raphson method. *SIAM Review*, 37(4):531–551, 1995.

[88] Guomin Zhang, Jianping Yin, En Zhu, and Ling Mao. On the selection of multi optimal imaging frames in single time slot for earth observation satellite. In *Young Computer Scientists, 2008. ICYCS 2008. The 9th International Conference for*, pages 791 –796, nov. 2008.

[89] E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, ETH Zurich, Switzerland, 1999.

[90] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In K. C. Giannakoglou et al., editors, *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, pages 95–100, Athens, Greece, 2001. International Center for Numerical Methods in Engineering.

[91] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257 –271, nov 1999.

[92] Eckart Zitzler and Simon Künzli. Indicator-based selection in multiobjective search. In Xin Yao et al., editors, *Parallel Problem Solving from Nature - PPSN VIII*, volume 3242 of *Lecture Notes in Computer Science*, pages 832–842. Springer Berlin Heidelberg, 2004.

[93] Eckart Zitzler, Marco Laumanns, and Stefan Bleuler. A tutorial on evolutionary multiobjective optimization. In *Metaheuristics for Multiobjective Optimisation*, pages 3–38. Springer-Verlag, 2003.