Robust Radiotherapy Appointment Scheduling

Farnaz Hajipour

A Thesis
in
The Department
of
Mechanical and Industrial Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Science in Industrial Engineering at
Concordia University
Montreal, Quebec, Canada

Decembre 2016

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: Farnaz Haji Pour

Entitled: Robust Radiotherapy Appointment Scheduling

and submitted in partial fulfillment of the requirements for the degree of

Master of Science in Industrial Engineering

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final Examining Committee:

Dr. Rajamohan Ganesan Chair

Dr. Daria Terekhov Examiner

Dr. Ketra Schmitt Examiner

Dr. Masoumeh Kazemi Zanjani Supervisor

Approved by _____

Chair of Department or Graduate Program Director

_____2016 _____

Dean of Faculty

Abstract

Robust Radiotherapy Appointment Scheduling

Farnaz Haji Pour

Optimal scheduling of patients waiting for radiation treatments is a quite challenging operational problem in radiotherapy clinics. Long waiting times for radiotherapy treatments is mainly due to imbalanced supply and demand of radiotherapy services, which negatively affects the effectiveness and efficiency of the healthcare delivered. On the other hand, variations in the time required to set-up machines for each individual patient as well as patient treatment times make this problem even more involved. Efficient scheduling of patients on the waiting list is essential to reduce the waiting time and its possible adverse direct and indirect impacts on the patient. This research is focused on the problem of scheduling patients on a prioritized radiotherapy waiting list while the rescheduling of already booked patients is also possible. The aforementioned problem is formulated as a mixed-integer program that aims for maximizing the number of newly scheduled patients such that treatment time restrictions, scheduling of patients on consecutive days on the same machine, covering all required treatment sessions, as well as the capacity restriction of machines are satisfied. Afterwards, with the goal of protecting the schedule against treatment time perturbations, the problem is reformulated as a cardinality-constrained robust optimization model. This approach provides some insights into the adjustment of the level of robustness of the patients schedule over the planning horizon and protection against uncertainty. Further, three metaheuristics, namely Whale Optimization Algorithm, Particle Swarm Optimization, and Firefly Algorithm are proposed as alternative solution methods. Our numerical experiments are designed based on a case study inspired from a real radiotherapy clinic. The first goal of experiments is to analyze the performance of proposed robust radiotherapy appointment scheduling (ASP) model in terms of feasibility of schedule and the number of scheduled patients by the aid of Monte-Carlo simulation. Our second goal is to compare the solution quality and CPU time of the proposed metaheuristics with a commercial solver. Our experimental results indicate that by only considering half of patients treatment times as worst-case scenario, the schedule proposed by the robust RAS model is feasible in the presence of all randomly generated scenarios for this uncertain parameter. On the other hand, protecting the schedule against uncertainty at the aforementioned level would not significantly reduce the number of scheduled patients. Finally, our numerical

results on the three metaheuristics indicate the high quality of their converged solution as well as the reduced CPU time comparing to a commercial solver.

Acknowledgement

I would like to thank my family, especially my sweetheart mother and my beloved father, and also my brother for supporting me spiritually to pave this way. This accomplishment would not have been possible without them.

I would like to express my sincere gratitude to my supervisor, Dr. Masoumeh Kazemi Zanjani, for her professional guidance and continuous encouragement throughout my thesis work. It has been my great honor and privilege to work under her supervision. She consistently allowed this research to be my own work but steered me in the right the direction whenever she thought I needed it.

Last but not the least, I would like to thank all my friends and officemates who supported and helped me with their knowledge and experience.
Thank you.

Table of Contents

Table of Figures

Table of Tables

Chapter 1: Introduction

As healthcare costs increase rapidly in developed countries and the demand for health services and the patients' expectations of service quality grows, providing efficient health systems would be inevitable. In other words, healthcare decision makers seek to more effective healthcare plans while facing with scarce resources and multiple stakeholders who often have conflicting goals to provide timely access to quality care for all patients. These complexities have made healthcare systems to become an attractive application area for operations research (OR). Various decision support techniques, developed based on OR methodologies and solution algorithms, are widely used in healthcare systems that provide the opportunity to reduce costs simultaneously and improve access to healthcare services (Ahmadi-Javid et al. 2016).

Outpatient medical centers such as radiotherapy clinics have become more central in healthcare systems in recent years due to the emphasis on preventive medical practices, shorter hospital stays, and more services being provided on an outpatient basis. An appropriate appointment system, as an essential component of efficient care delivery in outpatient clinics, enables such centers to deliver care at the right time, utilize medical resources optimally, and maximize patient and physician satisfaction (Cayirli & Veral 2003).

Radiotherapy is a way for treating many kinds of cancer, aiming at destroying a tumor or stopping its growth, and also relieving pain. In the radiation process, admitted patients undergo a set of clinical examinations, which allows oncologist assess their pathological condition. Based on the evaluation done by the oncologist, the total amount of radiation to be delivered is determined by the radiotherapist who fractionates this total amount to determine the dose fraction of the radiation to be given at each treatment session. Afterward, the treatment plan is defined based on the number of sessions patients are required to be treated each week.

After determining the treatment plan, the simulation phase occurs in order to setup all parameters related to the plan, such as linear accelerators (linacs) features. Linacs are special electrical devices that concentrate in beams and accelerate the emission of subatomic particles. The treatment scheduling takes place after the simulation phase, where the patients are assigned to machines, days and shifts assuring that all the sessions required are done in consecutive days and without any interruptions.

In this study, we confine our attention to an offline appointment scheduling system over a given planning horizon, where the maximum number of patients with the highest priorities on a waiting list of radiotherapy treatment must be assigned to specific days and shifts on a particular linac. This problem is formulated as a Binary-Integer-program (BIP) that assures: *i*) patients are scheduled based on several time restrictions such as release dates, due dates, and latest starting date; *ii*) patients are scheduled in consecutive days on the same machine such that the total number of prescribed sessions are covered; and *iii*) the schedule does not exceed the capacity of linacs on each day. Further, the model incorporates the list of previously booked patients on a certain linac, where the patients can be rescheduled on the days and shifts that they are available. This will provide more flexibility in scheduling the maximum number of non-booked patients over the planning horizon. It is worth noting that the proposed RAS model in this thesis extends the model in (Conforti et al. 2010) by incorporating more time restrictions (e.g., release and due dates) that must be taken into consideration while treating patients in such clinics.

Radiotherapy appointment scheduling (RAS) is featured with uncertain set-up times, treatment times, patients' availability (cancellation/no-show), and linacs availability (machine breakdown). Such uncertainties not only could increase the waiting time for scheduled patients but also might lead to underutilization of linacs. There are two approaches to deal with uncertain parameters in optimization models namely robust optimization (RO) (Ben-Tal & Nemirovski 1998; Bertsimas & Sim 2004) and stochastic programming (SP) (Birge and Louveaux 2011). While RO method seeks an optimal solution that is feasible in the presence of all outcomes of uncertain factors, SP approach revolves around defining proper corrective actions in the presence of various uncertain outcomes such that the expected performance of the system is optimized. Although stochastic programming is less conservative in comparison with robust optimization, it requires exact information on the probability distribution of uncertain parameters. Further, reformulating the RAS problem as a stochastic program with recourse would lead to a large-scale BIP model that is difficult to solve in real-size instances.

In this thesis, with the goal of protecting the schedule against treatment time perturbations, the aforementioned problem is reformulated as a cardinality-constrained robust optimization model (Bertsimas & Sim 2004). More precisely, the RO model finds an optimal schedule that is feasible for all outcomes of uncertain treatment times under a given budget of uncertainty. This approach

provides the possibility to control the degree of conservatism of the robust solution in the sense that a trade-off between the cost and degree of robustness can be obtained through choosing an appropriate budget of uncertainty.

The proposed RAS models, both in deterministic and uncertain contexts, are large-scale BIP's that are hard to solve for long planning horizons (i.e., more than one week) and large number of paitients on a waitng lists in a real-size radiotherapy clinic with a large number of linacs. Hence, we propose three metaheuristics, namely Whale Optimization Algorithm (WOA), Particle Swarm Optimization (PSO), and Firefly Algorithm (FA) to alleviate the computational complexity in the aforementioned cases.

Finally, a radiotherapy patient scheduling case study inspired from real data provided in (Conforti et al. 2010) is carefully designed for validating proposed deterministic and robust optimization models as well as metaheuristics. Further, we conduct an extensive set of Monte-Carlo simulation experiments in order to better analyze the impact of budget of uncertainty on the feasibility and cost of the schedule. In other words, by considering various budgets of uncertainty and several uncertainty sets, modeled as uniform intervals with different variances, we verify the feasibility and cost of schedules proposed by the RO and deterministic models.

The main contributions of the thesis, thus, revolve around *i*) extending the existing RAS models by including more time restrictions; *ii*) incorporating uncertain traetmnet times into the RAS problem and formulating it as a cardinality-constrained robust optimization model; *iii*) proposing three metaheuristics (i.e., WOA, PSO, and FA) to aleviate the complexity of the RAS model for large instances in terms of size of the waiting list, clinic, and length of planning horizon; and *iv*) conducting Monte-carlo simulation experimnets on a case study, carefully designed based on real data in the literature, so as to validate the robustness of proposed schedule in a realistic context

The rest of thesis is organized as follow. The detailed review of relevant literature is presented in Chapter 2. Chapter 3 provides a brief description of cardinality-constrained robust optimization approach and the three metaheuristics (WOA, PSO, and FA). In chapter 4, we present the problem description and formulation in both deterministic and uncertain contexts. The details of implementing metaheuristics on RAS problem is summarized in chapter 5. The experimental results are presented in chapter 6 while chapter 7 concludes the thesis and provides insights into future avenues of research.

Chapter 2: Literature review

In this chapter, we first provide a brief literature review on outpatient appointment scheduling. Then we narrow down into the existing literature on radiotherapy appointment scheduling problem which is directly related to the problem investigated in this thesis. The chapter is concluded by highlighting the current gaps in the literature.

2.1. Outpatient appointment scheduling

Healthcare systems can be approached either on an inpatient or an outpatient basis. An "Inpatient" system refers to the procedure by which the patient is admitted to the hospital primarily so that he/she can be closely monitored during the procedure and afterward, during recovery; whereas in an "Outpatient" system there is no need for hospital admission and treatment might be performed outside of the hospital. On the other hand, the environment of these systems can be divided into three categories: 1) primary care clinics, 2) specialty clinics, and 3) surgery clinics (Gupta, Diwakar; Denton 2008).

According to (Cayirli & Veral 2003), outpatient services are becoming a central component of health care because of an emphasis on pre-emptive medical practices and the shorter length of stay in hospitals.

A review of the literature on outpatient appointment systems, especially appointment scheduling, can be found in (Cayirli & Veral 2003; Gupta, Diwakar; Denton 2008; Ahmadi-Javid et al. 2016). In (Cayirli & Veral 2003), the primary goal was to review prior formulations, performance criteria used to evaluate appointment systems, classification of appointment systems studied in the literature, and analysis of methodologies adopted for appointment scheduling in outpatient services; whereas (Gupta, Diwakar;Denton 2008) focused on describing the most prevalent types of health care delivery systems with particular attention on the factors that make appointment scheduling conflicting and challenging.

Very recently, (Ahmadi-Javid et al. 2016) introduced a broader framework for categorizing outpatient appointment scheduling models based on 3 groups of strategic, tactical, or operational decisions. According to the authors, main strategic decisions incorporate access policy, number of servers, policy on acceptance of walk-in patients, and types of scheduling. As for tactical-level decisions, they refer to allocation of capacity to different patient groups, appointment intervals,

appointment scheduling window, block size, number of appointments per consultation session, panel size, and priority of patient groups. And finally among the most studied operational decisions, we can mention allocation of patients to servers, appointment day, appointment time, patient acceptance/rejection, patient selection from waiting list, and patient sequence. The authors also discussed the most common problem formulations, solution methods, and environmental factors in the outpatient appointment systems literature. Among such environmental factors, we can refer to patient unpunctuality, physician lateness, interruptions, patient no-show and cancellation, patient preferences, random service time, patient heterogeneity, and type of appointment required by patients. Inspired by abovementioned decision levels, this thesis focuses on the operational level.

2.2. Radiotherapy appointment scheduling

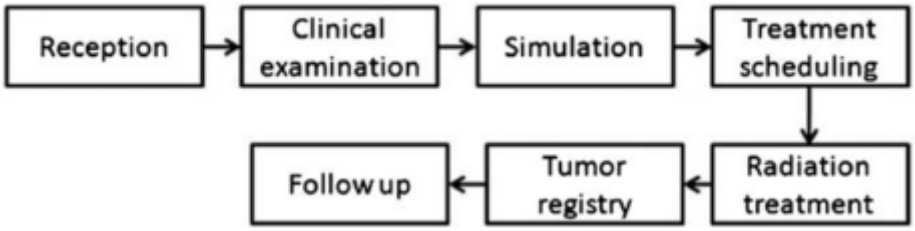
In the outpatient scheduling context, particular attention has been given to radiotherapy treatments (Conforti et al. 2009). Managing and scheduling patients in radiotherapy clinics is a quite challenging issue. According to Conforti et al. (2010), due to large dimension of the waiting lists in most cases, the time between the submissions of a radiotherapy request to the initiation of the delivery of the ionising radiation dose is typically long which potentially harms patient both directly (tumor growth) and indirectly (psychological distress for the patient). Therefore, there is a need for optimal patient scheduling based on reliable quantitative approaches. A review of radiotherapy appointment scheduling (RAS) problems and open research questions related to this topic can be found in Kapamara et al. (2006). In this section, we first provide an overview of radiotherapy treatment process. Afterward, we elaborate on various radiotherapy scheduling approaches investigated in the literature of RAS.

2.2.1. Overview of radiotherapy treatment process

Radiotherapy is a way for treating different kinds of cancer. It is often performed to destroy a tumor and to cure cancer. In this regard, it is defined as curative radiotherapy to provide long-term benefits to the patient. The radiotherapy may also be given before surgery to shrink cancer cells or after surgery to stop the growth of remaining ones. On the other hand, it can be deployed before, during, or after chemotherapy to improve treatment result. Sometimes, when it is not possible to cure cancer, the radiotherapy is used to relieve pain (Conforti et al. 2009).

The radiotherapy treatment process is complex and patients must undergo a set of steps in the pre-treatment stage before starting the treatment (Burke et al. 2011). These steps are as depicted in Fig. 1 (Conforti et al. 2010). According to the authors, after admission, the radiation oncologist carries out a clinical examination to assess the pathological conditions and evaluate the radiotherapy treatment. In this case, the total amount of radiation to be delivered and the dose fractions which will be delivered in the course of the treatment sessions during the planned time horizon are determined by a radiotherapist. This amount depends on the site, size and type of cancer, and pathological conditions of the patient.

Figure 1- Radiotherapy treatment process [adopted from Conforti et al. (2010)]



The procedure called fractionating the dose is a process in which a significant amount of radiation is delivered to a tumor safely over a time span of several weeks and allows to save healthy tissue from damage and gives it time to recover (Kapamara & Petrovic 2009). Afterward, the simulation phase takes place to set up all the parameters relevant to the most effective radiation treatment. In this phase, patient anatomical and pathological conditions and linear accelerator (linac) configuration must be carefully considered (Kapamara et al. 2006). After completing all phases preceding the treatment planning, the oncologist assigns a priority value to the patient based on the "severity" of pathological conditions. According to (Conforti et al. 2010) the literature on incorporating the priority of patients into radiotherapy appointment scheduling is scarce. Burke et al. (2011) classified patients under three different categories of emergency, urgent or routine. (Conforti et al. 2010) on the other hand, introduced the following four categories for assessing the priority for treatment: Priority A: emergency radiation treatment; priority B: curative radiation treatment; priority C: palliative and other radical radiation treatments; and priority D: combined chemotherapy and radiation treatment. According to the same authors, the treatment plan usually starts many days after the simulation phase, since it is essential to verify the current validity of the

parameter values specified at the time of simulation. In order to ensure that the therapy destroys as many of the cancer cells as possible, while avoiding adverse effects on healthy cells, a suitable "break", defined as a certain number of days, between two consecutive weeks, during which no treatments are delivered, should also be planned. Depending on a treatment plan and dosage fraction, the patient has to visit the treatment center several times along a week, for a number of weeks (Conforti et al. 2008). The length of the treatment depends on different factors and is very variable. In general, each treatment session takes from 5 to 15 minutes. Some other criteria required to estimate the actual treatment time has been reviewed in (Conforti et al. 2010). In the first treatment session, the linear accelerator is set up manually, whereas in the following sessions all parameters are automatically setup. Therefore, the first treatment session generally needs more time than the subsequent ones (Conforti et al. 2008; Conforti et al. 2010; Conforti et al. 2009).

It is worth to mention that the treatment is generally performed in an outpatient setting, where patients visit the hospital on a daily basis during their treatment period (Conforti et al. 2010). If an immediate booking is not possible, the patient is added into a waiting list. Therefore, the waiting list is partitioned into ordered sub-lists, in which patients with the same assigned priority are assigned to each sub-list. In many clinics, even with overbooking, there are more patients on the waiting list than the available capacity. In such cases, the patients are selected to be served based on various criteria, such as the patients' priority level and waiting time. In some studies, it is assumed that the priority assigned to each patient does not change until the patient is served. However, in reality, the priorities change in case the condition of the tumor could become more severe due to long waiting time (Kolisch & Sickinger 2008). Also, radiotherapy clinics face the challenge of selecting who will be served next when demand comes from emergency patients in the waiting list as well as pre-scheduled patients. To conclude, the main requirements that should be taken into account in scheduling the radiotherapy appointments encompass:

- The number of treatment sessions prescribed by the oncologist is fixed and has to be carried out on consecutive days in each week;
- Each patient should be delivered only one treatment session per day;
- the same linac must be used during the entire treatment plan since technical characteristics could vary among different machines;

- The capacity of each machine, given by the number of working hours (minutes), must not be exceeded on any given day;
- Each patient cannot start treatment before the release date in which the pre-treatment is finished (Castro & Petrovic 2012).
- The treatment should be completed before the due date assigned to each patient. This due date indicates the day by which the treatment sessions have to be finished.
- Patients usually prefer a specific day and physician over other days and service providers. These preferences often differ from one patient to another and may change over time. Also service providers might have various preferences (Gupta, Diwakar; Denton 2008).
- In reality, linacs have different performances, and each linac is used for a specific type of radiation (Burke et al. 2011)

2.2.2. Scheduling approach

There are two scheduling approaches in RAS problems: online and offline. In the online approach, patients are scheduled immediately upon the arrival of their request, while in the offline approach appointments are scheduled after all requests have arrived. In this section we classify the literature into online and offline scheduling approach separately.

2.2.2.1. Online radiotherapy appointment scheduling

Some decisions in an online scheduling, such as allocation of patients to servers, appointment day and time, are dynamic and determined during the continuous patient call-in process. The Markov decision process is one of the most useful tool for dealing with dynamic behavior of online systems. For instance, in (Kolisch & Sickinger 2008), this approach is used to model a system with multiple identical servers in which patients are selected from a waiting list in each period. They dynamically modeled the radiology departments of two German university medical centers with more than 1,200 and 1,400 beds, respectively by only considering the CT-devices. A CT (Computed Tomography) device makes use of computer-processed combinations of many X-ray images taken from different angles to produce cross-sectional (tomographic) images of specific areas of a scanned tissue, allowing the physician to see inside it without cutting. The goal is to allocate the available resources dynamically to the patients of the groups such that the expected total reward including revenues, waiting costs, and penalty costs is maximized. The authors also considered no-shows for inpatients, random arrival for outpatients and emergency as uncertain

parameters into the problem. It is worth mentioning that no-shows refer to when scheduled patients do not show up for their appointments. This is one of the major problems that almost all radiotherapy clinics are confronted with. Patients' no-show behavior reduces system efficiency and provider productivity by wasting medical resources. They also adopted two ways to manage the negative effects of no-show: 1) booking extra patients beyond the facility's capacity, which is called over-time; and 2) reducing appointment intervals.

Sauré et al. in (Sauré et al. 2012) developed an infinite-horizon Markov decision process (MDP) the dynamic multi-priority patient scheduling problem by introducing multiple appointment requests, multiple session durations and allowing parts of the appointments to be delivered using overtime. In order to deal with an intractable number of states and actions, they first transform their MDP model into its equivalent linear programming formulation. Then, the linear model is solved using column generation and uncertain parameters (i.e., number of sessions, session durations, wait time penalties, and demand rates) are validated by exploiting simulation.

Another frequently applied method in the radiotherapy online scheduling is simulation-based optimization that combines optimization and simulation modeling approaches to facilitate the search procedure in stochastic and online complex systems. For instance, (Kapamara et al. 2007) investigated the treatment process and identified bottlenecks from the interactions between patients (i.e., random arrivals) and resources (i.e., number of staffs, working hours,doctor availability, and machine breakdowns) by means of discrete-event simulation. They simulated the radiotherapy system of the University Hospitals of Coventry and Warwickshire (UHCW) in England by including additional radiotherapy services such as brachytherapy and unsealed sources therapy and extending the work in (Proctor et al. 2007). (Kapamara et al. 2007) and (Petrovic et al. 2009) proposed a daily scheduling approach which considered both the radiotherapy pre-treatment and treatment stages together.

Ogulata et al. in (Ogulata et al. 2009) proposed a slack capacity approach to minimize delays in treatments posed by potential lengthening in current patients treatments and to maintain efficient use of daily treatment capacity in a radiation oncology department. According to this approach, some part of the daily patient capacity is reserved as a slack (unused) capacity in order to prevent the treatment delays. They also conducted a simulation analysis of the scheduling approach in

order to assess the proposed approach under different environmental conditions and to determine appropriate scheduling policy parameter values.

(Liang et al. 2014) simulated the Department of Hematology and Oncology in Lahey Hospital (USA). They considered no-show probabilities, unpunctual arrivals, and also the uncertainty of service time with some realistic distributions for modeling heterogeneous (patient dependent / service dependant) service times. Unpunctual arrival is defined as the difference between a patient appointment time and actual arrival time.

The optimization models proposed for online scheduling in (Pérez et al. 2016) were formulated based on the data provided by the Scott & White Healthcare Clinic Nuclear Medicine Department (USA) with different type of resources. They assumed each patient follows an ordered sequence of examinations, referred as a pathway, which is determined based on the classification of the patient and site of the tumor. Along with a sequence, different resources (staff and machines) are also required to be assigned to each patient. Moreover, they are faced with a situation in which a patient requires a resource more than once, referred as recirculation, and resource concurrence when an operation requires more than one resource simultaneously. In this situation, resources are not continuously available throughout the scheduling horizon since their schedules are partially filled with other appointments. They developed heuristic algorithms (namely, fixed resource (FR) and procedure resource assignment (PRA)) as well as simulation models for scheduling nuclear medicine patients and resources. Both algorithms schedule patients following a general scheduling structure. The algorithms first search for the patient's preferred day for the appointment. If the search results in an appointment for which the patient has to wait more than a month, then an earlier appointment on an alternate day is considered.

Some studies used the results obtained from the offline case to examine the online case. for instance, (Pérez et al. 2013) formulated the problem of scheduling patients and resources in nuclear medicine clinics in three forms: offline, online, and stochastic online. The offline model is formulated by using integer linear programming and assigns patients to a specific server, step, and time slot. In the online model, patient requests are not known in advance; rather, they arrive sequentially one at a time and are scheduled as they arrive. At each specific time the offline model is used to decide how to schedule requests received at that time by fixing all the other patients already scheduled. Stochastic online model is formulated using two-stage stochastic programming.

The proposed stochastic online scheduling model is similar to the online one, except that possible future requests are also taken into account when scheduling patients. The goal is to find the best day and time to accommodate the current request such that the expected performance measures are improved.

(Marie-andr & Rousseau 2015) investigated online scheduling adopted from offline scheduling approach for patients in a radiotherapy clinic in Cancer Centre of Laval, Canada. The offline system is modeled using integer programming in order to assign patients to appointment days and time slots. The stochastic online system is formulated by exploiting two-stage stochastic programming approach. In this model, the first stage decision variables are appointment times and the second stage variables are auxiliary variables such as the patients' waiting times, server's idle time, and system's overtime.

2.2.2.2. Offline radiotherapy appointment scheduling

Offline scheduling systems are more popular than the online ones in the literature. The reason may be that these systems are easier to model than the online systems. Moreover, as the electronic appointment scheduling systems are rapidly developing, the importance of offline scheduling is also growing. In such systems, the appointment requests are collected via an IT tool (e.g., email ore web-based portal) over a specific time period; afterwards, patients can be informed of their appointment time, according to the offline scheduling approach.

Almost all deterministic models in the offline scheduling systems are formulated by using integer linear programming. Deterministic models are often used to formulate problems that are less affected by the uncertainty caused by random arrivals and random treatment times. Typical complications for these models include capacity and due date constraints with multi-resource and multi-stage treatment procedures. For instance, Burke et al. (2011) modeled a system in which patients are assigned to a numbers of the linacs with different performances while appointment day is also to be determined. They assumed that each machine could emit a specific type of radiation; hence, the patients should be assigned to an appropriate linac according to the specific radiation they require.

Castro & Petrovic (2012) modeled the appointment time scheduling of radiotherapy pre-treatment, in which each patient follows a specific sequence of operations. Hence, the decisions are to assign patients to servers, steps, day, and time. They also adopted a multi-objective solution procedure

by developing some dispatching rules aimed at minimizing the weighted number of patients exceeding the waiting time targets, maximum lateness, and sum of weighted lateness.

Conforti et al. (2010) considered only one group of identical machines for scheduling of patients in a waiting list both in terms of appointment day and time. The strategy adopted into their model is based on a non-block system. Basically, the radiotherapy scheduling strategies adapted in the literature can be categorized into two systems: block and non-block. In the former, the workday is split into a fixed number of time blocks, which usually are same in duration (typically 10 - 15 minutes). In the non-block system, different treatment times are assigned to patients. It is important to observe that the use of uniform appointment blocks, adopted in many radiotherapy centers, is in general a poor real workload representation becouase the effective treatments can take either more or less time than the specified time block. This means that the total surplus of time assigned to each scheduled patient over the working day could be used for scheduling other patients waiting for treatment. Thus, the non-block scheduling strategy turns to be more efficient than the block one. The Model in (conforti et a. 2010) reschedules some patients, who have a treatment plan in progress, only when it is necessary by exploiting their availabilities. This provides the possibility of scheduling urgent patients.

Dynamic disruption/disturbances compound the complexity of RAS problems while affect the choice of solution approches. One way of responding to the disturbances is the rescheduling approach. Rescheduling can be interpreted in two ways: 1) only the time slots could change with respect to the last planned week, and 2) it is possible to delay the day of the first weekly session. Frequent revision is necessary for better scheduling results although it is not always beneficial to reschedule after every unexpected event. This helps to further define RAS as a patient scheduling/rescheduling problem. In these situations, appointment times of most pre-scheduled patients are postponed and the RAS systems must decide which pre-scheduled and which urgent patients should be seen based on available resources; or the system must make decisions about rescheduling pre-scheduled patients and how to manage new requests during the disruption period.

Conforti et al. in (Conforti et al. 2008; Conforti et al. 2009) proposed models for offline scheduling of patients in a single server radiotherapy clinic by adopting block strategy. The models proposed in (Conforti et al. 2008; Conforti et al. 2009) provide the possibility to reschedule booked patients in order to improve system's performance through booking patients with higher priorities. The

decisions in their models involve appointment day and time for patients on the waiting list. Authors in (Conforti et al. 2009) extended the work in (Conforti et al. 2008) by taking into account the time constraints such as due dates and release dates. The due date reffers to the date by which the treatment sessions of the patient have to be completed. Whereas, release date defines the first day after the pre-treatment stage is completed and the patient is available/ready to be treated.

Heuristics form an important class of inexact solution methods to solve radiotherapy patients scheduling problems. Although these methods do not guarantee optimal solutions, sometimes they are indispensable in practical or real-size problems because of the presence of complex environmental characteristics and stochastic factors. Metaheuristic algorithms are among other inexact solution methods for integer programming problems. Although metaheuristics can help to solve problems with more realistic assumptions or in less computational time, there are a limited number of studies that have employed them for solving RAS models. For instance, (Petrovic & Leite-Rocha 2008), proposed four constructive approaches namely target approach, utilisation threshold approach, schedule creation day, and maximum number of day in advance for radiotherapy scheduling. They also developed a GRASP (Greedy Randomized Adaptive Search Procedure)-based algorithm for improving the solution obtained by the constructive approaches. The target approach operates in a forward (backward) procedure from the release date (due date) of each patient, while trying to schedule the required number of sessions subject to the given constraints. If it is not possible to accommodate all the required sessions, the algorithms move the start day forward (backward) and tries again. In the utilisation threshold approach, a threshold of machine utilisation is defined for each priority of patients. In the third constructive approach, schedule creation day approach, specific days of the week are selected for each patient priority whenever a schedule can be created. If a patient arrives on a day when it is not specified to create a schedule for that priority, the schedule will be created on the first following allowed day. The Maximum number of days in Advance approach introduces waiting times (days) before creating a schedule for the patient after he/she has arrived. In this way, it is possible to reserve more space for the patients of higher priorities in the earlier dates. The GRASP-based algorithm has two phases; in the first phase, patients are ordered lexicographically in the same way as in the constructive approaches; while in the second phase, after the initial solution has been constructed, a local search is applied.

(Kapamara & Petrovic 2009) proposed an algorithm for scheduling patients in a radiotherapy department while undergoing numbers of steps with multiple servers aimed at minimizing the weighted lateness. An initial solution is created using a rule-based heuristic; then the hill climbing method is used to improve this initial solution.

(Petrovic et al. 2011) applied multi-objective Genetic Algorithm (GA) models to generate schedules for radiotherapy patients at Arden Centre Cancer, Coventry, UK. They considered different sets of identical linacs in their mathematical models formulated for the scheduling and sequencing of patients. Due to the stochastic nature of the GAs and uncertainty in the daily number of newly arrived patients and their categories, they used the GA to generate schedules for 10 different sets of daily arrived patients and repeated it 10 times for each set of patients with different initial populations. Three GAs are developed and implemented which treat radiotherapy patient categories, namely emergency, palliative and radical patients in different ways: *i*) Standard-GA, which considers all patient categories equally, *ii*) Weighted-GA, which operates with different weights assigned to the patient categories and *iii*) KB-GA, which has an embedded knowledge on the scheduling of emergency patient category.

(Kapamara et al. 2006) formulated RAS as a job shop scheduling problem and reviewed exact and metaheuristic approaches for solving similar job shop scheduling problems.

2.3. Summary of relevant literature

In this chapter, we presented a comprehensive review of analytic studies on RAS problems. The aim of this work was to provide an overview on appointment scheduling in radiotherapy clinics and identifying the aspects that have received limited attention in operation research literature. We summarize some of major findings as follows.

Tables (1) and (2) classify reviewed papers according to different criteria. Table (1) categorizes current literature into operational decisions investigated, type of scheduling, uncertain factors, objective function(s), modeling approaches adopted, and solution methodologies. Whereas, in Table (2), the literature is categorized into the problem settings such as number of servers, number of steps, stage (i.e., pre-treatment and treatment), blocking strategy (i.e., block or non-block), time constraints, resource performance (i.e., identical resources or resource with same performances),

and rescheduling policy. Finally, Table (3) provides some additional definitions for the characteristics mentioned in Tables (1) and (2).

Our survey on the existing literature indicate that most analytic studies have neglected some time constraints that may further complicate mathematical models (i.e., lead time of operations in multi-step RAS, radio-pharmaceutical shelf life, breaks, due dates, and release dates). These restrictions appear in realistic problems; hence, considering them can increase the applicability of the resulting models.

Due to inherent uncertainty and sequential property of RAS decisions, stochastic programming, and Markov decision process have been used most frequently. In most presented models, the time-based measures have been used to evaluate system performance. However, to the best of our knowledge no studies have been done on adapting robust optimization approaches to protect the models against uncertain parameters perturbations.

With respect to solution methods, we noticed that most of the literature is directed toward heuristic methods that are easy to implement and have reasonable computation time for real-life problems. Recently, due to increased interest in modeling more realistic RAS, the mathematical models presented in the literature are difficult to solve optimally in an acceptable amount of time. Therefore, finding a good solution procedure in terms of quality and computation time for nearly realistic models is an important and challenging issue in this context.

In this thesis, we fill some of the aforementioned gaps by extending the non-block RAS model, proposed in (Conforti et al. 2010), where we incorporate more realistic time constraints (i.e., due dates and release dates of patients). Then, we propose the robust counterpart of the model to protect the schedule against perturbations in treatment times. Finally, we implement three metaheuristic algorithms namely Whale Optimization Algorithm (WOA), Particle Swarm Optimization (PSO), and Firefly Algorithm (FA) in order to solve the models.

Table 1- Characteristics of reviewed papers

Reference	Operational decisions	Type of scheduling	Uncertainties	Objective	Modeling approach	Solution methods
Burke et al. (2011)	day, time, server	offline		minimize the number of patients who miss the breach date+ weighted number of patients who miss the JCCO maximum acceptable target+ weighted number of patients who miss the JCCO good practice target+ weighted average squared waiting times	BIP	
Castro et al. (2012)	day, time, server, operation	offline		minimise the weighted number of patients exceeding the waiting time targets+ maximum lateness+ sum of weighted lateness	BIP	multi-objective by dispatching rules
Conforti et al. (2010)	day, time, server	offline		maximize the weighted number of new scheduled patients	BIP	CPLEX
Conforti et al. (2008)	day, time	offline		maximize the weighted number of new scheduled patients	BIP	LINGO
Conforti et al. (2009)	day, time	offline		maximize the weighted sum of booked and newly scheduled patients	BIP	CPLEX
Kapamara et al. (2009)	day, time, server, operation	offline		minimise the total weighted lateness for the received patients		tabu search, hill climbing heuristic
Kapamara et al. (2006)					Job Shop	meta heuristics
Kapamara et al. (2007)	server, operation	online	Number of staffs, working hours, Doctor availability, Machine breakdowns	minimise waiting time to first session		discrete-event simulation
Kolisch & Sickinger (2008)	time, server	online	No-shows for inpatients, random arrival for outpatients and emergency	maximize expected total reward consisting of revenues, waiting costs, and penalty costs	MDP	Linear Capacity Allocation, first come first served, random selection

Reference	Operational decisions	Type of scheduling	Uncertainties	Objective	Modeling approach	Solution methods
Perez et al. (2016)	time, server, step	online		minimize waiting time from the time of the procedure request until the time of the appointment+ maximize performance ratio (number of times patients are scheduled on the date requested above all patient requests) + minimize cycle time (time patient spends in the system) +maximize equipment utilization+ maximize human resource utilization+ maximize number of patients served per day	BIP	The fixed resource (FR) algorithm +The procedure resource assignment (PRA) algorithm+ Simulation
Petrovic et al. (2011)	time, server, step	offline	daily number of newly arrived patients, their categories and treatment plans	minimize average waiting time+ average tardiness of the patients		Standard-GA, KB-GA, Weighted-GA
Petrovic & Leite-Rocha (2008)	day	offline		minimize average weighted tardiness of patients		Target Approach Utilisation, Threshold Approach, Schedule Creation Day Approach, Maximum Number of Days in Advance Approach, and Grasp
Proctor et al. (2007)	time, server, operation	online	arrival (demand), resource capacity	minimize waiting time		simulation
Sauré et al (2012)	day, time	online	number of sessions, session durations, wait time penalties, demand rates	minimize penalties associated with the resulting patient wait times, the cost associated with the use of overtime, and the penalties associated with postponing some of the booking decisions	infinite-horizon Markov decision process and its equivalent LP	column generation + simulation for uncertainties

Reference	Operational decisions	Type of scheduling	Uncertainties	Objective	Modeling approach	Solution methods
Larsson (1993)	servers	offline				Macro
Liang et al. (2014)	time, server, step	online	unpunctual arrivals, stochastic service times and treatment durations	minimize patient waiting times + balance clinic workload		discrete-event simulation
Marie-andre & Rousseau (2015)	day, server	offline online	arrival of patients at the center	minimize cost of scheduling and overtime	MIP two-stage SP	rule-based: ASAP online: the greedy algorithm and the primal-dual algorithm
Ogulata et al. (2009)	day	online	patients arriving frequency, slack capacity	minimize the percentage of unaccepted patients, treatment delay, the quantity of the patients waiting in queue maximize normal capacity usage ratio, slack capacity usage ratio		Slack Capacity approach+ Discrete-event simulation
Perez et al. (2013)	time, server, step	offline online	future arrivals	maximize number of newly scheduled minimize total waiting time	BIP two-stage SP	stochastic online scheduling algorithm (heuristic)

Table 2- Characteristics of reviewed papers (problem settings)

Reference	Number of servers	Number of steps	Stage	Blocking Strategy	Time Constraints	Resources Performance	Reschedule Booked Patients
Burke et al. (2011)	Multiple	Single	treatment	non-block	Due dates, release dates	different	no
Castro et al. (2012)	Multiple	Multiple	pre-treatment	block	Due dates, release dates, lead time of operations, waiting time targets	different	no
Conforti et al. (2010)	Multiple	Single	treatment	non-block	latest starting date	same	yes
Conforti et al. (2008)	Single	Single	treatment	block	latest starting date		yes
Conforti et al. (2009)	Single	Single	treatment	block	latest starting date, due dates, release dates		yes
Kapamara et al. (2009)	Multiple	Multiple	pre-treatment and treatment	block		different	no
Kapamara et al. (2006)							
Kapamara et al. (2007)	Multiple	Multiple	pre-treatment	non-block		different	
Kolisch & Sickinger (2008)	Multiple	Single	treatment	block		same	no

Reference	Number of servers	Number of steps	Stage	Blocking Strategy	Time Constraints	Resources Performance	Reschedule Booked Patients
Larsson (1993)	Multiple	Single	treatment			different	no
Liang et al. (2014)	Multiple	Multiple	pre-treatment and treatment	non-block		same	no
Marie-andre & Rousseau (2015)	Multiple	Single	treatment	block (22 min)	deadline for first treatment or the latest starting date, release dates	same	no
Ogulata et al. (2009)	Single	Single	treatment	non-block			no
Perez et al. (2013)	Multiple	Multiple	pre-treatment and treatment	block (10 min)	imposed by the decay of the radio-pharmaceuticals		no
Perez et al. (2016)	Multiple	Multiple	pre-treatment and treatment	block (10 min)			no
Petrovic et al. (2011)	Multiple	Multiple	pre-treatment and treatment	non-block	release dates, due dates	different	no
Petrovic & Leite-Rocha (2008)	Multiple	Single	treatment		due dates, release dates	different (low, high, electron energy type)	no
Proctor et al. (2007)	Multiple	Multiple	pre-treatment and treatment	non-block		different	no
Sauré et al (2012)	Single	Single	treatment	block (10 or 12 min)		same	no

Table 3- Additional definitions

Operational decisions	allocating patients to servers		Deals with the problem of selecting which service provider should serve a particular patient. This category of decisions is involved in multi-server systems.
	appointment day scheduling		In RAS with multi-day scheduling horizon, the appointment day for each patient must be determined so that indirect waiting time for each patient is reasonable according to the patient's priority level.
	appointment time scheduling		Deals with finding a specific time when a patient is scheduled to start receiving care so that a performance criterion is optimized.
	allocation patients to operations/steps		In multi-step RAS, the pathway for each patient should be determined through operation allocation decisions.
Problem settings	Number of servers		Refers to how many servers there are to meet patient demand.
	Time constraints	Lead time of operations	This lead time is the latency between the demand initiation and execution of the operation.
		Latest starting date	Indicates the latest day by which the first treatment session could be planned.
		break	In order to guarantee that the therapy destroys as many of the cancerous cells as possible, avoiding negative effects on many healthy cells, a suitable ‘‘break” between sets of consecutive sessions should be also planned, defined as a certain number of days during which no treatments are delivered.
		shelf life of pharmaceuticals	As far as the radio-pharmaceuticals have specific shelf life, the putrefaction of these materials is highly probable and may pose lots of costs on the system. Hence, taking into account this sort of time restriction has large impact on the system efficiency.
Objective	JCCO waiting time targets		These targets have been established by the Joint Council for Clinical Oncology. They determine the good practice and the maximum acceptable waiting times from the date the patient is first visited for suspected cancers to the first session of treatment for each category of patients.
	breach dates		States that patients must start their treatment no later than 62 days from the date on which they are referred to an oncologist by their general physician and no later than 31 days from the date when the decision to treat with radiotherapy was made.
	lateness/tardiness		Is defined as the difference between the completion day of the pre-treatment phase and the due date.

Chapter 3: Methodology

In this chapter, we elaborate on the main methodology adopted to formulate RAS problem under uncertain treatment times. We also provide the detailed description of the three metaheuristics adopted to solve deterministic and robust RAS models.

3.1. Cardinality-constrained robust optimization approach

In this research, we implement the cardinality-constrained robust optimization approach proposed by Bertsimas and Sim (Bertsimas & Sim 2004) for linear programming problems. In what follows, we present the core idea and a summary of this method. Let consider the following uncertain linear programming (LP) model:

$Max\ Z = c^T x$

s.t.

$\sum_{j=1}^{n} \tilde{a}_j\, x_j \le b_i \quad \forall A_j \in K_j, j = 1, \dots, n$

$x \ge 0$ (1)

Consider the *i*th constraint of the nominal problem $\tilde{a}_i^T x \le b_i$. Let J_i be the set of coefficients that are subject to parameter uncertainty, i.e., $a_{ij}, j \in J_i$ take values in the interval $[\bar{a}_{ij} - \hat{a}_{ij}\,, \bar{a}_{ij} + \hat{a}_{ij}\,]$, where $\bar{a}_{ij}$ denotes the nominal (estimated) value of $\tilde{a}_{ij}$, and half-length $\hat{a}_{ij}$ measures the precision of the estimate. We define the scaled deviation η_{ij} of $\tilde{a}_{ij}$ from its nominal value as:

$$\eta_{ij} = \frac{\tilde{a}_{ij} - \bar{a}_{ij}}{\hat{a}_{ij}}$$

The scaled deviation of a parameter always belongs to [-1, 1].

Although the aggregate scaled deviation for constraint *i*, $\sum_{j=1}^{n} \eta_{ij}$, can take any value between –*n* and *n*, the fact that aggregate forecasts are more precise than individual ones suggests that the true values taken by $\sum_{j=1}^{n} \eta_{ij}$ will belong to a much narrower range. Intuitively, some parameters will

exceed their nominal values while others will fall below estimate, so the η_{ij} will tend to cancel each other out.

This approach protects against violation of constraint *i* deterministically, when only a pre-specified number Γ_i of coefficients changes. In other words, it guaranties that the solution is feasible if less than Γ_i uncertain coefficients change. Accordingly, for every constraint *i*, we introduce a parameter Γ_i (not necessarily integer) that take values in the interval $[0,|J_i|]$. The role of this parameter is to adjust the robustness of the proposed method against the level of conservatism of the solution. It is unlikely that all of the $\tilde{a}_{ij}, j \in J_i$ will change. The nature will be restricted in its behavior, in that only a subset of the coefficients will change in order to adversely affect the solution. The goal is to be protected against all changes up to $\lfloor \Gamma_i \rfloor$ of these coefficients are allowed to change by $\hat{a}_{it}$, and one coefficient a_{it} changes by $(\Gamma_i - \lfloor \Gamma_i \rfloor)\hat{a}_{ij}$.

We first consider the following nonlinear model to formulate the above situation, where the first constraint represents a protection function for each uncertain constraint *i*:

$Max\ Z = c^T x$

s.t. :

$$\sum_j \bar{a}_{ij} x_j + \max_{\{S_i \cup t_i | S_i \subseteq J_i, |S_i| = \lfloor \Gamma_i \rfloor, t_i \in J_i \setminus S_i\}} \left\{ \sum_{j \in S_i} \hat{a}_{ij}\, y_j + (\Gamma_i - \lfloor \Gamma_i \rfloor)\hat{a}_{it_i} y_t \right\} \leq b_i \quad \forall i$$

$$-y_j \leq x_j \leq y_j \ \forall j$$

$$l_j \leq x_j \leq u_j \ \forall j$$

$$y_j \geq 0 \ \forall j \qquad (2)$$

In order to reformulate model (2) as an LP we need the following propositions:

- ***Proposition 1***: Given a vector x^*, the protection function of the *i*th constraint

$$\beta_i(x^*, \Gamma_i) = \max_{\{S_i \cup t_i | S_i \subseteq J_i, |S_i| = \lfloor \Gamma_i \rfloor, t_i \in J_i \setminus S_i\}} \left\{ \sum_{j \in S_i} \hat{a}_{ij} \left| x_j^* \right| + (\Gamma_i - \lfloor \Gamma_i \rfloor)\hat{a}_{it_i} \left| x_j^* \right| \right\} \qquad (3)$$

equals the objective function of the following optimization problem:

$\beta_i(x^*, \Gamma_i) = max \sum_{j \in J_i} \hat{a}_{ij} |x_j^*| z_{ij}$

s.t.

$$\sum_{j \in J_i} z_{ij} \leq \Gamma_i$$

$$0 \leq z_{ij} \leq 1 \quad \forall j \in J_i \tag{4}$$

Proof: the optimal solution of (4) consist of $\lfloor \Gamma_i \rfloor$ variables at $\Gamma_i - \lfloor \Gamma_i \rfloor$. This is equivalent to the selection of subset $\{S_i \cup t_i | S_i \subseteq J_i, |S_i| = \lfloor \Gamma_i \rfloor, t_i \in J_i \backslash S_i\}$ with corresponding cost function $\sum_{j \in S_i} \hat{a}_{ij} |x_j^*| + (\Gamma_i - \lfloor \Gamma_i \rfloor) \hat{a}_{it_i} |x_j^*|$.

In order to formulate model (2) as an LP, we first consider the dual of problem (4) as follows:

$min \sum_{j \in J_i} p_{ij} + z_i \Gamma_i$

s.t.

$$z_i + p_{ij} \geq \hat{a}_{ij} |x_j^*| \quad \forall j \in J_i$$

$$p_{ij} \geq 0 \quad \forall j \in J_i$$

$$z_i \geq 0 \tag{5}$$

By strong duality theorem, since problem (4) is feasible and bounded for all $\Gamma_i \in [0, |J_i|]$, then the dual problem (5) is also feasible and bounded and their objective values coincide. Using proposition 1, we have that $\beta_i(x^*, \Gamma_i)$ is equal to the objective function of problem (5). Substituting to problem (10), we obtain:

$Max\ Z = c^T x$

s.t.:

$$\sum_j \bar{a}_{ij} x_j + z_i \Gamma_i + \sum_{j \in J_i} p_{ij} \leq b_i \quad \forall \mathrm{i}$$

$$z_i + p_{ij} \geq \hat{a}_{ij} y_j \quad \forall i, j \in J_i$$

$$-y_j \leq x_j \leq y_j \ \forall j$$

$$l_j \leq x_j \leq u_j \ \forall j$$

$$y_j \geq 0 \ \forall j$$

$$z_i \geq 0 \ \forall i$$

$$p_{ij} \geq 0 \quad \forall i, j \in J_i \tag{6}$$

3.2. Metaheuristic Algorithms

metaheuristics are formally defined as some iterative generation processes that guide a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space. Further, learning strategies are used to structure information in order to find efficiently near-optimal solutions (Osman and Kelly, 1996). Techniques that constitute metaheuristic algorithms range from simple local search procedures to complex learning processes. Such algorithms are approximate and usually non-deterministic. They may incorporate mechanisms to avoid getting trapped in confined areas of the search space. Metaheuristics are not problem-specific and may make use of domain-specific knowledge in the form of heuristics that are controlled by the upper level strategy. Today's more advanced metaheuristics use search experience (embodied in some form of memory) to guide the search.

Metaheuristics are becoming more and more popular in engineering applications because they: (*i*) rely on rather simple concepts and are easy to implement; (*ii*) do not require gradient information; (*iii*) can bypass local optima; and (*iv*) can be utilized in a wide range of problems covering different disciplines (Yang 2010a).

Most metaheuristic algorithms are nature-inspired as they have been developed based on some abstraction of nature. Nature-inspired metaheuristic algorithms solve optimization problems by mimicking biological or physical phenomena. Nature has evolved over millions of years and has found perfect solutions to almost all the problems she faced. We can thus learn the success of problem-solving from nature and develop nature-inspired heuristic and/or metaheuristic algorithms. More particularly, some nature-inspired algorithms are inspired by Darwin's evolutionary theory. Consequently, they are said to be biology-inspired or simply bio-inspired. Two major components of any metaheuristic algorithms are: selection of the best solutions and randomization. The selection of the best solutions ensures that the solutions will converge to optimality, while the randomization avoids the solutions being trapped at local optima and, at the same time, increase the diversity of the solutions. The good combination of these two components will usually guarentee that the global optimum is achieved (Osman & Kelly 1996).

Nature-based metaheuristic algorithms can be classified into many ways. One way is to classify them as: population-based and trajectory-based. Population-based (or agent-based) metaheuristics share a common feature regardless of their nature. The search process is divided into two phases: exploration and exploitation. The optimizer must include operators to globally explore the search space: in this phase, movements (*i.e.* perturbation of decision variables) should be randomized as much as possible. The exploitation phase follows the exploration phase and can be defined as the process of investigating in detail the promising area(s) of the search space. Exploitation hence pertains to the local search capability in the promising regions of decision space found in the exploration phase. Finding a proper balance between exploration and exploitation is the most challenging task in the development of any metaheuristic algorithm due to the stochastic nature of the optimization process. On the other hand, trajectory-based metaheuristics use a single agent or solution that moves through the decision space or search space in a piecewise style. A better move or solution is always accepted while a not-so-good move can also be accepted with certain probability. The steps or moves trace a trajectory in the search space with a non-zero probability that this trajectory can reach the global optimum (Yang 2010a).

Nature-inspired metaheuristic algorithms can also be grouped into three main categories: evolution-based, physics-based, and swarm-based methods. Evolution-based methods are inspired by the laws of natural evolution. The search process starts with a randomly generated population which is evolved over subsequent generations. The strength point of these methods is that the best individuals are always combined together to form the next generation of individuals. This allows the population to be optimized over the course of generations. The most popular evolution-inspired technique is Genetic Algorithms (GA) that simulates the Darwinian evolution (Mirjalili & Lewis 2016). Physics-based methods imitate the physical rules in the universe. The most popular algorithms are Simulated Annealing (SA), Gravitational Local Search (GLSA), and Big-Bang Big-Crunch (BBBC).

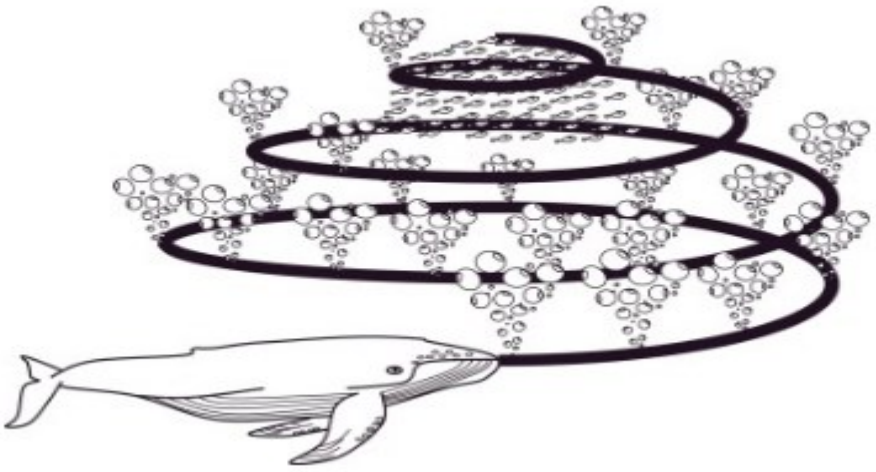
The third group of nature-inspired methods includes swarm-based techniques that mimic the social behavior of groups of animals. The most popular algorithm is Particle Swarm Optimization (PSO), originally developed by (Eberhart & Kennedy 1995). PSO is inspired by the social behavior of bird flocking or insect swarming. It uses a number of particles (candidate solutions) that fly around in the search space to find the best solution (*i.e.* the optimal position). Meanwhile, they all trace

the best location (best solution) in their paths. In other words, particles consider their own best solutions as well as the best solution the swarm has obtained so far. Another popular swarm-based algorithm is Firfly Algorithm (FA), first proposed by Yang (2010). This algorithm is inspired by the social behavior of firflies in a firefly colony. In fact, the social intelligence and flashing behaviour of fireflies to attract potential prey, and to attract mating partners is the main inspiration of this algorithm. The positions of fireflies are evolved over the course of iterations by which fireflies with lower light intensity move toward the brighter ones. Another swarm-based technique is whale optimization algorithm (WOA). This algorithm mimicks the hunting behavior of humpback whales, where a group of whales encircle the prey along a circle or a '9'-shaped path as shown in figure 2. The population of whales create distinctive bubbles called bubble net, so that each of them can update their position according to other whales' position closer to the prey. The main characteristics of this algorithm incorporate the simulated hunting behavior with random or the best search agent to chase the prey and the use of a spiral to simulate bubble-net attacking mechanism of humpback whales (Mirjalili & Lewis 2016).

Swarm-based metaheuristic methods started to be attractive since PSO was proven to be very competitive with evolution-based and physical-based algorithms. Generally speaking, swarm-based algorithms have some advantages over evolution-based algorithms. For example, they preserve search space information over subsequent iterations while evolution-based algorithms discard any information as soon as a new population is formed. They also include less operators compared to evolutionary approaches (e.g. selection, crossover, mutation, elitism, in sGA) and hence are easier to implement (Yang 2010a).

In what follow we discuss three major modern swarm-based metaheuristic methods applied in this thesis, including Whale Optimization Algorithm (WOA), Particle Swarm Optimization (PSO), and Firefly Algorithm (FA).

Figure 2- Bubble-net feeding behavior of humpback whales[adopted from (Mirjalili & Lewis 2016)]



3.2.1. Whale Optimization Algorithm

Whale Optimization Algorithm (WOA) mimics the social behavior of humpback whales. The algorithm is proposed by (Mirjalili & Lewis 2016) inspired by the bubble-net hunting strategy. Humpback whales prefer to hunt a school of krill or small fishes close to the surface. This is done by creating distinctive bubbles along a circle. It is worth to mention that a whale in this algorithm corresponds to one feasible solution and the prey refers to the optimal solution. Humpback whales encircle the prey within a shrinking mechanism and along a spiral-shaped path simultaneously; therefore, each whale is assumed to choose between the shrinking mechanism or the spiral-shaped method to update its position by probability of 50%. The mathematical formulation of the aforementioned hunting strategies is as follows:

$$\vec{X}(t+1) = \begin{cases} \overrightarrow{X^*}(t) - \vec{A} \,.\, \vec{D} & if\ p < 0.5 \\ \overrightarrow{D'} \,.\, e^{bl} \,.\, \cos(2\pi l) + \overrightarrow{X^*}(t) & if\ p \geq 0.5 \end{cases} \tag{7}$$

Where $\vec{X}(t+1)$ denotes updated position for the next iteration ($t + 1$) and p is a random number in [0, 1]. In what follows, we introduce all parameters and variables in (7). First the shrinking mechanism formulation ($p < 0.5$) is disscussed, afterward we focuse on the parameters and variables of spiral-shaped mechanism formaulation ($p \geq 0.5$). In the shrinking mechanism formulation ($p < 0.5$), $\overrightarrow{X^*}(t)$ denotes a random whale chosen from current population if the algorithm is in the exploration phase ($|\vec{A}| > 1$). On the other hand, it stands for the best feasible solution obtained so far if the algorithm is in the exploitation phase ($|\vec{A}| < 1$). It is worth

mentioning that in the shrinking mechanism, the variation of the $\vec{A}$ vector can be exploited to choose between exploration phase (Search for prey) and exploitation phase (Bubble-net attacking method). In fact, in exploration phase, humpback whales search randomly according to the position of each other. Therefore, $\vec{A}$ is used as the random values greater than 1 or less than −1 to force each search agent (whale) to move, respectively, toward or far away from a reference whale. The idea is to search different regions of the search space (i.e. feasible set). The position of a search agent (whale) in the exploration phase is updated according to a randomly chosen search agent (whale) instead of the best position found so far. Thus $|\vec{A}| > 1$ emphasizes exploration and allow the WOA algorithm to perform a global search (see Figure 3.a). The formulation is as follows:

$$\vec{D} = |\vec{C} \,.\, \overrightarrow{X_{rand}} - \vec{X}| \tag{8}$$

$$\vec{X}(t+1) = \overrightarrow{X_{rand}} - \vec{A} \,.\, \vec{D} \tag{9}$$

Where $\overrightarrow{X_{rand}}$ is a random agent (whale) chosen from the current population and $\vec{X}$ is the current position of the whale. The vectors, $\vec{A}$ and $\vec{C}$ are calculated as follows:

$$\vec{A} = 2\,\vec{a} \,.\, \vec{r} - \vec{a} \tag{10}$$

$$\vec{C} = 2 \,.\, \vec{r} \tag{11}$$

Where $\vec{a}$ is linearly decreased from 2 to 0 over the course of iterations and $\vec{r}$ is a random vector in [0, 1].

On the other hand, in the exploitation phase ($|\vec{A}| < 1$), Humpback whales can recognize the location of prey and encircle them to hunt. Based on this behaviour, the optimal position in the search space is not known a priori. Thus, it is assumed that the current best candidate solution (position of agent $\overrightarrow{X^*}(t)$) is the optimum one or is close to the optimum. After the best search agent (whale) is determined, the other search agents (whales) will then try to update their positions towards it (see Figure 3.b). The following equations demonstrate this behavior:

$$\vec{D} = |\vec{C} \,.\, \overrightarrow{X^*}(t) - \vec{X}(t)| \tag{12}$$

$$\vec{X}(t+1) = \overrightarrow{X^*}(t) - \vec{A} \,.\, \vec{D} \tag{13}$$

Where *t* is the current iteration, $\vec{A}$ and $\vec{C}$ are coefficient vectors discussed previously, $X^*(t)$ is the best solution obtained so far, and $\vec{X}(t)$ is the current position vector for the search agent. It is worth to mention that, $X^*(t)$ should be updated in each iteration once a better solution is obtained for the population's position.

In the spiral-shaped mechanism ($p \geq 0.5$ in (7)), the distance between the current position ($X(t)$) and incumbent position ($X^*(t)$) is first calculated. A spiral equation is then created between the current position of the whale and the prey to mimic the helix-shaped movement of humpback whales (see figure 4) as:

$$\vec{X}(t+1) = \overrightarrow{D'} \cdot e^{bl} \cdot \cos(2\pi l) + \overrightarrow{X^*}(t) \tag{14}$$

Where $\overrightarrow{D'} = \left|\overrightarrow{X^*}(t) - \vec{X}(t)\right|$ and indicates the distance of the current position to the best one so far, b is a constant for defining the shape of the logarithmic spiral, and l is a random number in $[-1, 1]$.

To conclude, the WOA algorithm starts with a set of random feasible solutions as initial position of the whales from the population. At each iteration, depending on the value of p, WOA is able to switch between a spiral and shrinking movement. In the shrinking mechanism, search agents (whales) update their positions with respect to either a randomly chosen position (exploration phase) or the best position obtained so far (exploitation phase). Finally, the WOA algorithm is terminated by the satisfaction of a termination criterion, for instance the maximum number of iterations.

Figure 3- Shrinking mechanism; a) Exploration phase, b) Exploitation phase [adopted from (Mirjalili & Lewis 2016)]

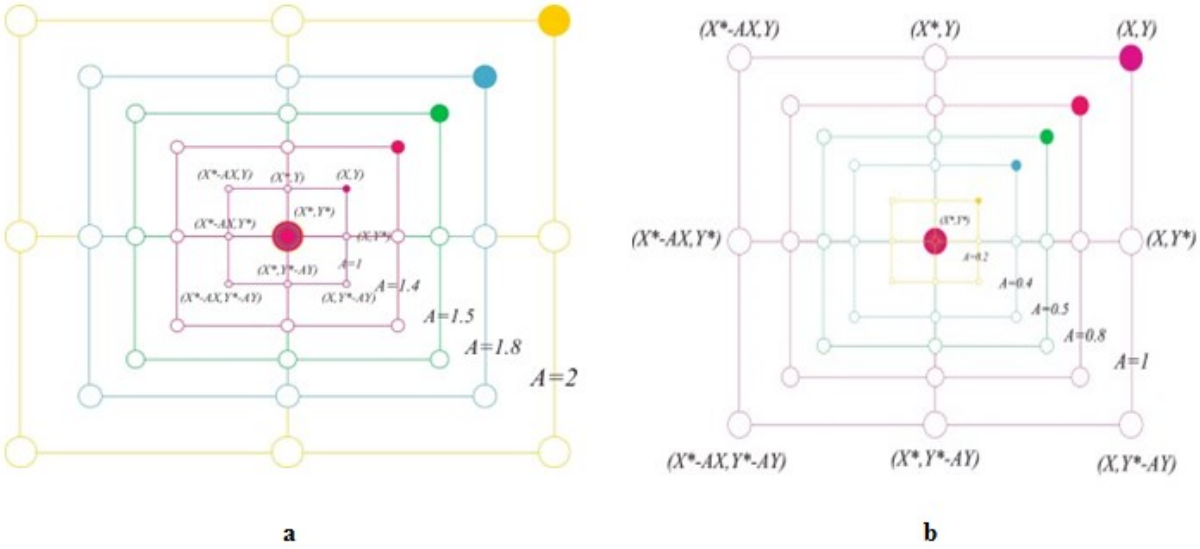
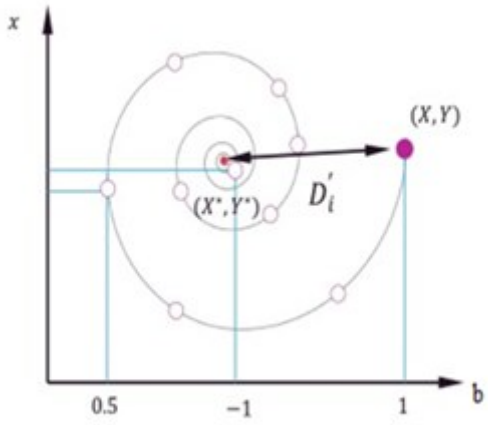


Figure 4- Spiral-shaped mechanism [adopted from (Mirjalili & Lewis 2016)]



The pseudo code of the WOA algorithm is presented in *Algorithm 1*. It is worth mentioning that, index permutation matrices represent the members of population. In the RAS problem, each representing schedule as will be further elaborated in chapter 5 (section 5.1). I_{max} refers to the population size and t_{max} denote the maximum number of iterations. Also, the fitness function is described in *Algorithm 2*.

Algorithm 1- WOA

1. Generate initial index permutation matrices for the members of population (*pop (i)*) randomly;

2. For i=1 to I_{max}

    2.1. *pop (i) = fitness (pop (i));*

    2.2. *bestpop* = best population;

3. while ($t_{max} < 15$)

    3.1. for each index permutation matrix

    3.2. Update *a, A, C, l,* and *p* according to equations (10) and (11);

    3.3. if ($p < 0.5$)

        3.3.1. if ($|A| < 1$)

            3.3.1.1. Update the matrices of the current population by the Eq. (8- 9);

        3.3.2. else if ($|A| \geq 1$)

            3.3.2.1. Select a random matrix;

            3.3.2.2. Update the matrices of the current population by the Eq. (12- 13);

        3.3.3. end

    3.4. else if ($p \geq 0.5$)

        3.4.1. Update the matrices of the current population by Eq. (14);

    3.5. end

    3.6. end

    3.7. Check if any matrix goes beyond the search space and amend it (divide whole row by ceil of maximum element of the row);

    3.8. *Pop (i) = fitness (pop (i));*

    3.9. *bestpop*= best population*;*

    3.10. $t = t + 1$;

4. end

5. return best pop;

Algorithm 2- fitness function

1. Get index permutation matrix;
2. For all permutation of indices in the obtained order in the index permutation matrix
   2.1. If all constraints are satisfied
      2.1.1. Assign 1 to decision variables; otherwise, assign 0 to them.
3. Calculate the objective function and return it;

3.2.2. Particle Swarm Optimization

Particle swarm optimization (PSO) is a metaheuristic developed by (Eberhart & Kennedy 1995) inspired from swarm intelligence. It is based on the bird and fish flock movement behavior. While searching for food, the birds are either scattered or go together before they locate the place where they can find the food. While the birds are searching for food from one place to another, there is always a bird that can smell the food very well. In other words, the bird is perceptible of the place where the food can be found, while having the better food resource information. Since they are transmitting the information, especially the good information at any time while searching the food from one place to another, the birds will eventually flock to the place where food can be found. As far as particle swam optimization algorithm is concerned, the solution swam is compared to the bird swarm; i.e., the birds' moving from one place to another is equivalent to the development of the solution swarm; good information corresponds to the best solution; and the food resource is equivalent the optimal solution during the whole process. The optimal solution can be worked out in particle swarm optimization algorithm by the cooperation of each individual bird (feasible solution).

In the particle swarm optimization algorithm, the swarm consists of "*n*" particles, and the position of each particle corresponds to the potential solution in the search space. Each particle changes its position according to the following three principles: *i*) to keep its inertia; *ii*) to change the condition according to the swarm's best position; and *iii*) to change the condition according to its best position. The position of each particle in the swarm is affected both by the best position during its movement (individual experience) and the position of the best particle in its surrounding (near

experience). Hence, each particle can be shown by its current velocity (v_i) and position (x_i), the best position of each individual, and the best position of the surrounding (see Figure 5). The velocity and position of each particle change according the following equations :

$$v_i^{t+1} = \omega\, v_i^t + \varphi_p\, r_p\, (p_i - x_i^t) + \varphi_g\, r_g\, (g - x_i^t) \quad (15)$$

$$x_i^{t+1} = x_i^t + v_i^t \quad (16)$$

Where v_i^{t+1} is the updated velocity vector of particle i in the next iteration ($t+1$). v_i^t is current velocity vector for particle i; x_i^t is current position of particle i; p_i represents the best position of the individual i at its best position; g is the quantity of the swarm at its best position; r_p and r_g are two random vectors taking values between 0 and 1. The parameter ω is the velocity importance coefficient; φ_p and φ_g are the learning parameters or acceleration constants regulating the length when flying to the best particle of the whole swarm and to the best individual particle, respectively. If the constant is too small, the particle is probably far away from the target field; while if it is too big, the particle will maybe fly to the target field suddenly or fly beyond the target field. The proper constants can control the velocity of the particle's flying. These constants can typically be taken as $\omega \approx 1$ *and* $\varphi_p \approx \varphi_g \approx 2$.

The initial locations of all particles should distribute relatively uniformly so that they represent most regions of the search space. The initial velocity of a particle can be set as zero, that is, $v_i^0 = 0$. Although v_i can take any values, it is usually bounded in some range $[0, v_{max}]$ · Particles' velocities in each dimension are clamped to a maximum velocity v_{max}.

Figure 5- Vector representation of PSO [adapted from (Eberhart & Kennedy 1995)]

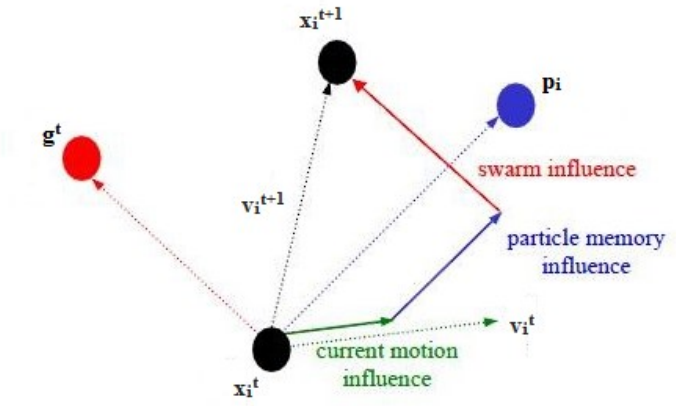


Figure 5 is a vector representation of PSO velocity and position updates in a two-dimensional space. At each iteration t, a particle x_i^t updates its velocity and position according to equations (14) -(15).

The pseudo code of the PSO algorithm is presented in *Algorithm 3*. It is worth mentioning that index permutation matrices represent the members of population (i.e., a schedule in RAS problem). These matrices are further discussed in chapter 5 (section 5.1). I_{max} refers to the population size and t_{max} is the maximum number of iterations.

Algorithm 3- PSO

1. For each particle $i = 1, ..., I_{max}$ do:
   1.1. Initialize the particle's position (index permutation matrix) with a uniformly distributed random vector: $x_i \sim U(0, 1)$
   1.2. Initialize the particle's best known index permutation matrix to its initial one: $p_i \leftarrow x_i$
   1.3. If (*fitness* (p_i) > *fitness* (g)) update the swarm's best known matrix: $g \leftarrow p_i$
      1.3.1. Initialize the particle's velocity: v_i
2. Until a termination criterion is met (t_{max}), repeat:
   2.1. For each particle $i = 1, ..., I_{max}$
      2.1.1. generate the uniformly distributed random parameters: $r_p, r_g \sim U(0, 1)$
      2.1.2. Update the particle's velocity by: $v_i \leftarrow \omega\, v_i + \varphi_p\, r_p\, (p_i - x_i) + \varphi_g\, r_g\, (g_d - x_i)$
      2.1.3. Update the particle's position (matrix) by following equation: $x_i \leftarrow x_i + v_i$
      2.1.4. If (*fitness* (x_i) > *fitness* (p_i)) do:
         2.1.4.1. Update the particle's best known position (matrix): $p_i \leftarrow x_i$
      2.1.5. If (fitness (pi) > fitness (g))
         2.1.5.1. update the swarm's best known position (matrix): $g \leftarrow pi$
3. end for
4. return *g* as the best solution.

3.2.3. Firefly Algorithm

The flashing light of fireflies is a stupendous sight in the summer sky of the tropical regions. There are lots of firefly species, and most of them produce short and rhythmic flashes. Two main

functions of such flashes are to attract potential prey, and mating partners (communication). Inspired by this social behaviour of fireflies, Firefly Algorithm (FA), developed by (Yang 2010b), starts with a random initial population of fireflies as initial feasible solutions to an optimization model. At each iteration of this algorithm, after checking the light intensity of each pair of fireflies their positions are updated in such a way that fireflies with lower light intensity are attracted to the ones with higher light intensity. The new population is accordingly obtained and sorted based on the light intensity and the firefly with the highest light intensity is selected as the optimal solution.

The light intensity at a specific distance *r* from the light source follows the inverse square law. In other words, the light intensity *I* decreases as the distance *r* increases ($I \propto \frac{I}{r^2}$).

In the Firefly Algorithm (FA) the following three idealized rules are used:

- All fireflies are unisex. Thus, one firefly attracts other fireflies regardless of their sex;
- Attractiveness is associated with their brightness, thus for any pair of flashing fireflies, the less bright one will move towards the brighter one. The attraction of each pair decrease as their distance increases;
- The brightness of a firefly is determined or affected by the objective function perspective. In other words, the flashing signal can be formulated in such a way that it is related to the objective function to be optimized. For a maximization problem, the light intensity can simply be commensurate with the objective value.

There are two important issues in firefly algorithm to address: the light intensity variation and attractiveness formulation. For simplicity, it can always be assumed that the attractiveness of a firefly is determined by its light intensity which is associated with the objective function.

In the simplest case for maximization problems, the brightness *I* of a firefly at a specific location *x* can be chosen as $I(x) \propto f(x)$. However, the attractiveness β is relative, it should be judged by other fireflies or seen in the eyes of the beholder. Therefore, it will change with the distance between firefly *i* and firefly *j* (r_{ij}). In addition, light intensity decreases with the distance from its source, and light is also absorbed in the air, so the attractiveness should be allowed to change with the absorption scale.

In the simplest form, the light intensity *I(r)* varies according to the inverse square law:

$$I(r) = \frac{I_s}{r^2} \tag{17}$$

Where I_s *is* the source intensity. For a given environment with a fixed light absorption coefficient γ, the light intensity I varies with the distance r. That is

$$I = I_0 e^{-\gamma r} \tag{18}$$

Where I_0 is the initial light intensity. Avoiding the singularity at $r = 0$ in the expression $\frac{I_s}{r^2}$, the combined effect of both the inverse square law and absorption can be approximated as the following Gaussian function

$$I(r) = I_0 e^{-\gamma r^2} \tag{19}$$

Since the firefly's attractiveness is proportional to the light intensity seen by adjacent fireflies, the attractiveness β of a firefly can be defined as

$$\beta = \beta_0 e^{-\gamma r^2} \tag{20}$$

Where β_o is the attractiveness at r = 0. As it is often faster to calculate $\frac{1}{(1+r^2)}$ than an exponential function, the above function, if necessary, attractiveness can conveniently be approximated as

$$\beta = \frac{\beta_0}{1+\gamma r^2} \tag{21}$$

Equations (19) - (21) define a characteristic distance $\Gamma = \frac{1}{\sqrt{\gamma}}$ over which the attractiveness changes significantly as the distance is increased (i.e. from β_o to $\beta_0 e^{-1}$ in equation (20) or $\frac{\beta_0}{2}$ in (21)).

In the actual implementation, the attractiveness function $\beta(r)$ can be any monotonically decreasing function such as the following generalized form

$$\beta(r) = B_0 e^{-\gamma r^m}, \ (m \geq 1) \tag{22}$$

The distance between any pair of fireflies i and j at x_i and x_j, respectively, is the Cartesian distance

$$r_{ij} = \|x_i - x_j\| = \sqrt{\sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2} \tag{23}$$

Where $x_{i,k}$ is the kth element of the spatial coordinate x_i of ith firefly. In 2-D case, we have

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{24}$$

The movement of the attracted firefly *i* to another more attractive (brighter) firefly *j* is determined by:

$$x_i = x_i + \beta_0 e^{-\gamma r_{ij}^2}\left(x_j - x_i\right) + \alpha\epsilon_i \quad (25)$$

Where the second term in (25) is due to attraction; while the third term is due to randomization with α being the randomization parameter, and ϵ_i is a vector of random numbers drawn from a Gaussian or uniform distribution. For most of our implementation, we can take $\beta_o = 1$ and $a \in [0,1]$.

It is worth pointing out that the movement equation is a random walk tendentious to the brighter fireflies. If $\beta_o = 0$, the equation becomes a simple random walk. Furthermore, the randomization term can easily be extended to other distributions. The parameter γ specifieses the variation of the attractiveness, and its value is difinitely important in determining the speed of the convergence and how FA behaves. In theory, $\gamma \in [0, \infty)$, but in practice, γ is determined by the characteristic length (Γ) of the system to be optimized. Thus, in most applications, it typically varies between 0.1 and 10. For a fixed γ, the characteristic length becomes

$$\Gamma = \gamma^{-1/m} \quad (26)$$

And as $m \to \infty$, $\Gamma \to 1$. Hence, from (25), for a given length scale Γ in an optimization problem, the parameter γ can be initialized as $\gamma = \frac{1}{\Gamma^m}$.

The pseudo code can be summarized as *Algorithm 4*. It is worth mentioning that, index permutation matrices are the members of population each representing a schedule in RAS problem. These matrices are discussed in chapter 5 (section 5.1). I_{max} refers to the population size (or the number of particles) and t_{max} is the maximum number of iterations.

Algorithm 4- FA

1. Define absorption coefficient γ ;
2. Generate initial index permutation matrices for the members of population (*pop (i)*) randomly;
3. For *i=1 to* I_{max}
   3.1. pop (i) = fitness (pop (i));
4. While ($t < t_{max}$)
   4.1. for *i = 1 : n* (all *n* matrixes)
      4.1.1. for *j = 1 : n* (*n* matrixes)
         4.1.1.1. if *fitness (j) > fitness(i)*
            4.1.1.1.1. Move matrix (firefly) *i* towards *j* by: $x_i = x_i + \beta_0 e^{-\gamma r_{ij}^2}\left(x_j - x_i\right) + \alpha\epsilon_i$;
            4.1.1.1.2. Evaluate new matrices and update the objective value (light intensity);
         4.1.1.2. end if
      4.1.2. end for *j*
   4.2. end for *i*
   4.3. Rank matrices and find the current best;
5. end while
6. return the current best firefly

Chapter 4: Problem description and formulation

In this section, we first present the deterministic mathematical model for RAS problem; then we develop its robust counterpart in section 4.2.1.

4.1. Deterministic appointment scheduling model in radiotherapy clinics

4.1.1. Assumptions and Notations

In what follows, we describe the details on the specific radiotherapy appointment scheduling problem that allows maximizing the number of newly scheduled patients, chosen from patients waiting to start their treatment (waiting list). Given a set of already booked patients (in the previous planning horizon), the aim is to schedule patients when at least one linear accelerator (linac) is available, ensuring that all treatment sessions specified for each patient in pre-treatment phase, will be carried out. We extend the model proposed by (Conforti et al. 2010) by considering time constraints (i.e. release dates and due dates). The due date refers to the date by which the treatment sessions of the patient have to be completed. Whereas, release date defines the first day after the pre-treatment stage is completed and the patient is available/ready to be treated. The scheduling policy is based on priority categories assigned by the physician based on the pathological conditions evaluated at the beginning of each planning horizon.

We first report the following notations used in mathematical models:

- K, set of weekdays in the planning horizon;
- F, set of available shifts in each workday;
- M, set of linacs (machines) with same performance;
- BP, set of patients that have already started the treatment (booked patients);
- WP, waiting list of patients waiting to start the radiotherapy plan (waiting patients);
- T_{mkf} , capacity of machine m during shift f of day k (minutes).

We first remark the validity of following assumptions:

- The planning horizon is considered as six consecutive weekdays (from Monday to Saturday), then we assume that $|K| = 6$. The schedule is generated weekly, generally on Saturday; in this way, the patients have sessions fixed in advance.
- Each patient should be assigned only to one shift in a day.
- Booked patients are available to be re-scheduled and assigned to the same machine at least in one known shift.
- During the first treatment session, a set-up stage is carried out before the radiation in order to validate the parameters fixed during the simulation phase.
- The treatment sessions have to be performed within consecutive days. There is no preemption in the treatment plan.
- Patients in WP are previously sorted in decreasing priority values. Similar priorities are also sorted by arrival time (FIFO basis).
- The modeling procedure is ‘‘offline” with respect to the radiotherapy treatments.

For all patients $p \in BP$ and $j \in WP$, the following parameters are available:

- t_p (t_j), the number of consecutive days required in weekly treatment sessions.
- ld_p (ld_p), latest day by which the patient can start the t_p (t_j) weekly therapy sessions.
- r_p (r_j) , release day by which the patient is ready to start the treatment sessions.
- d_p (d_p), due day by which the patient has to finish the weekly treatment sessions.

Moreover, other parameters required to schedule patients belonging to WP and BP are reported in what follows.

4.1.1.1. Parameters of patients in WP

- p_{rj}, priority value assigned by the oncologist based on the ‘‘severity” of the tumor’s condition.
- $\tilde{s}_{mj}$, radiation duration of patient j on the machine m.
- set^1_{mj}, set-up time for patient j on machine m during the first session of the treatment plan. This value is evaluated during the simulation phase.
- set_{mj}, set-up time for treating the patient j on machine m, in the sessions following the first one.

- $s_{mj}^{1} = set_{mj}^{1} + \tilde{s}_{mj}$, time of the first treatment of the treatment plan for patient *j* on machine *m*.
- $s_{mj} = set_{mj} + \tilde{s}_{mj}$, time of the treatments following the first one for patient *j* on machine *m*.
- w_j, the weight assigned to patient *j*. Let $|WP|$ be the number of patients belonging to *WP*, the values of the weight w_j are determined by setting $w_{|WP|} = a$; $w_{|WP|-1} = b$; and $w_j = w_{j+1} + w_{j+2} + (|WP| - j)\ \forall j = 1, \dots, |WP| - 2$, where a and b are non-negative integer values and typically small. Hence, we obtain a decreasing sequence that allows to discriminate among patients with the same priority value and the same number of treatment sessions on the basis of the entering time to the waiting list *(WP)*.

As mentioned earlier, we assume that those patients that have already started the treatment plan in the past (booked patients) can be rescheduled by taking into account their availability during the shifts on each day. Also, such patients must be treated on the same machine started before for the entire treatment plan. It is worth noting that booked patients refer to those patients who require several non-consecutive weeks of treatment sessions and by the time of scheduling during the new planning horizon (current week) have already completed a certain number of treatment weeks. Nevertheless, their previously booked sessions during current week are subject to change. This provides more flexibility in scheduling patients on the waiting list.

4.1.1.2. Parameters related to booked patients

- lpt_p, last weekday of the last planned treatment session.
- $break_p$, possible interval (in days) between two consecutive planned weeks.
- $delay_p$, possible delay (in days) in starting the first weekly treatment.
- s_{mp}, the treatment time of patient *p* assigned to machine *m*.
- sd_p, the starting day during which the first treatment session of the current planning week is delivered. In reality, when booking a patient *p*, the ‘‘break” between two consecutive planned weeks has to be taken into account.

$$sd_p = \begin{cases} break_p - |K| + lpt_p & if\ break_p > |K| - lpt_p \\ 1 & Otherwise \end{cases}$$

- lsd_p , the latest weekly starting day refer to the latest day the patient p can start the weekly treatment. In some cases a patient $p \in BP$ could delay the weekly starting day by the maximum delay ($delay_p \geq 0$) days, assuring the completion of t sessions.

$$lsd_p = min\{sd_p + delay_p, |K| - t_p + 1\}$$

- $av_{pkf} = \begin{cases} 1, & if\ patient\ p\ is\ available\ to\ be\ treated \\ & in\ shift\ f\ o\ day\ k \\ 0, & otherwise \end{cases}$

the decision variables are described as follows:

- $\bar{Z}_{pkf} = \begin{cases} 1, & if\ the\ first\ treatment\ for\ patient\ p\ in\ BP \\ & is\ carried\ out\ during\ shift\ f\ on\ day\ k \\ 0, & otherwise \end{cases}$
- $\bar{y}_{pkf} = \begin{cases} 1, & if\ patient\ p\ in\ BP\ is\ treated\ during \\ & shift\ f\ on\ day\ k \\ 0, & otherwise \end{cases}$
- $z_{mjkf} = \begin{cases} 1, & if\ the\ first\ treatment\ for\ patient\ j\ in\ WP \\ & is\ carried\ out\ during\ shift\ f\ on\ day\ k\ and\ machine\ m \\ 0, & otherwise \end{cases}$
- $y_{mjkf} = \begin{cases} 1, & if\ patient\ j\ in\ WP\ is\ treated\ on\ machine\ m \\ & during\ shift\ f\ on\ day\ k \\ 0, & otherwise \end{cases}$
- $X_{mj} = \begin{cases} 1, & if\ the\ patient\ j\ in\ WP\ is\ processed\ on \\ & machine\ m \\ 0, & otherwise \end{cases}$

This binary variable has been introduced with the aim to simplify formulating certain conditions.

4.1.2. Model formulation

$$\text{Max} \sum_{j=1}^{|WP|} \sum_{m=1}^{|M|} w_j X_{mj} \quad (27)$$

Subject to

$$\sum_{m=1}^{|M|} X_{mj} \leq 1 \qquad \forall j \in WP \ (28)$$

$$X_{mj} = \sum_{k=1}^{ld_j} \sum_{f=1}^{|F|} Z_{mjkf} \qquad \forall j \in WP\ \forall m \in M \ (29)$$

$$\sum_{m=1}^{|M|} \sum_{k>ld_j}^{|K|} \sum_{f=1}^{|F|} Z_{mjkf} = 0 \qquad \forall j \in WP \ (30)$$

$$\sum_{f=1}^{|F|} y_{mjkf} \leq X_{mj} \qquad \forall m \in M \ \forall j \in WP \ \forall k \in K \quad (31)$$

$$\sum_{m=1}^{|M|} \sum_{k=1}^{r_j-1} \sum_{f=1}^{|F|} Z_{mjkf} = 0 \qquad \forall j \in WP \ (32)$$

$$\sum_{m=1}^{|M|} \sum_{k=d_j+1}^{|K|} \sum_{f=1}^{|F|} y_{mjkf} = 0 \qquad \forall j \in WP \ (33)$$

$$\sum_{f=1}^{|F|} (y_{mjkf} + Z_{mjkf}) \leq X_{mj} \qquad \forall j \in WP \ \forall k \in K \ \forall m \in M \ (34)$$

$$\sum_{d=k+1}^{k+t_j-1} \sum_{f=1}^{|F|} y_{mjdf} \geq (t_j - 1) \sum_{f=1}^{|F|} Z_{mjkf} \qquad \forall m \in M \ \forall j \in WP \ \forall k \in K: k \leq ld_j \ (35)$$

$$\sum_{k=1}^{|K|} \sum_{f=1}^{|F|} y_{mjkf} + \sum_{k=1}^{ld_j} \sum_{f=1}^{|F|} z_{mjkf} = t_j X_{mj} \qquad \forall m \in M \ \forall j \in WP \ (36)$$

$$\sum_{j=1}^{|WP|} (S_{mj}^{1} Z_{mjkf} + S_{mj} y_{mjkf}) + \sum_{p=1}^{|BP|} S_{mp} av_{pkf} \bar{y}_{pkf} \leq T_{mkf} \qquad \forall m \in M \ \forall k \in K \ \forall f \in F \ (37)$$

$$\bar{y}_{pkf} \leq av_{pkf} \qquad \forall p \in BP \ \forall k \in K \ \forall f \in F \ (38)$$

$$\sum_{k=1}^{lsd_p} \sum_{f=1}^{|F|} av_{pkf} \bar{Z}_{pkf} = 1 \qquad \forall p \in BP \ (39)$$

$$\bar{y}_{pkf} \geq \bar{Z}_{pkf} \qquad \forall p \in BP \ \forall k \in K \ \forall f \in F \ (40)$$

$$\sum_{k=1}^{|K|} \sum_{f=1}^{|F|} av_{pkf} \bar{y}_{pkf} = t_p \qquad \forall p \in BP \ (41)$$

$$\sum_{d=k+1}^{k+t_p-1} \sum_{f=1}^{|F|} av_{pdf} \bar{y}_{pkf} \geq (t_p - 1) \sum_{f=1}^{|F|} \bar{Z}_{pkf} \qquad \forall p \in BP \ \forall k \in K: k \leq ld_p \ (42)$$

$$\sum_{f=1}^{|F|} av_{pkf} \bar{y}_{pkf} \leq 1 \qquad \forall p \in BP \ \forall k \in K \ (43)$$

$$\sum_{k=d_p+1}^{|K|} \sum_{f=1}^{|F|} \bar{y}_{pkf} = 0 \qquad \forall p \in BP \ (44)$$

$$\sum_{k=1}^{r_p-1} \sum_{f=1}^{|F|} \bar{Z}_{pkf} = 0 \qquad \forall p \in BP \ (45)$$

$$y_{mjkf}, Z_{mjkf} \in \{0,1\} \qquad \forall m \in M \ \forall j \in WP \ \forall k \in K \ \forall f \in F \ (46)$$

$$X_{mj} \in \{0,1\} \qquad \forall m \in M \ \forall j \in WP \ (47)$$

$$\bar{y}_{pkf}, \bar{Z}_{pkf} \in \{0,1\} \qquad \forall p \in BP \ \forall k \in K \ \forall f \in F \ (48)$$

Detailing the structure of the model, we first remark that the objective function (27), to be maximized, is a weighted sum of newly scheduled patients chosen from the waiting list, where the weights represent patients' priorities.

To ensure that each newly scheduled patient is assigned only to one machine during the entire planning horizon, the following constraints are formulated:

$$\sum_{m=1}^{|M|}\sum_{k=1}^{ld_j}\sum_{f=1}^{|F|} Z_{mjkf} \leq 1 \; \forall j \tag{49}$$

By introducing the auxiliary variables $X_{mj,}$ constraints (28) are obtained from (49). These constraints, together with constraints (29) and (30), assure that only one machine is assigned to each newly scheduled patient, and only one first treatment session of the entire planning horizon is booked. Constraints (31) guarantee that each treatment session, following the first one, can be assigned to the same machine as the one used in the first session, only to one shift per day. To assure that exactly *t* treatments will take place in consecutive weekdays for each newly scheduled patient on the same assigned machine during the entire planning horizon, constraints (34)– (36) are formulated. Indeed, a new session will be booked only and only if the available machine time per shift, in each day, is greater than the duration required for the patient, as formulated by constraints (37). The treatment sessions in consecutive weekdays, without interruption, have to be also ensured for patients belonging to *BP* who have already started their treatment. Therefore, taking into account the availability, given by av_{pkf}, each session is booked only when the patient is available, as formulated by constraints (38). Constraints (39) fix the first treatment day in the considered planning horizon, whereas constraints (40) imply that the first treatment is a booked session. In other words, constrains (40) assign the first session ($\bar{Z}_{pkf}$) from generally booked sessions ($\bar{y}_{pkf}$). Constraints (41) and (42) ensure that the number of scheduled treatments is exactly equal to the number of required treatments t_p and in consecutive days. Since each patient can undergo exactly one treatment per day, constraints (43) are introduced. Constraints (32), (33), (44) and (45) are the time constraints that assure the session cannot be assigned before release dates and after the due dates for both types of patients. Finally, constraints (46) – (48) define the domains of decision variables.

4.2. Radiotherapy appointment scheduling under uncertainty

The RAS problem is affected by three sources of uncertainty namely random duration of the first treatment, the treatment times following the first one for patients on the waiting list and treatment time of booked patients. Notice that these uncertain parameters affect constraints (37) in model (27)-(48). In this study, we model the aforementioned treatment times as uncertain intervals. Let's denote $\tilde{s}_{mj}^{1}$ as the uncertain time of the first treatment for patients on the waiting list with the nominal value of $\bar{s}_{mj}^{1}$, $\tilde{s}_{mj}$ as uncertain time of treatments following the first one with a nominal value of $\bar{s}_{mj}$, and $\tilde{s}_{mp}$ as uncertain treatment time for booked patients with a nominal value of $\bar{s}_{mp}$. $\tilde{s}_{mj}^{1}$ is assumed symmetric and takes values in the interval $[\bar{s}_{mj}^{1} - \hat{s}_{mj}^{1}, \bar{s}_{mj}^{1} + \hat{s}_{mj}^{1}]$. Then, the scaled deviation a_{mj} (belonging to [−1, 1]) of $\tilde{s}_{mj}^{1}$ from its nominal value is defined as $a_{mj} = \frac{\tilde{s}_{mj}^{1} - \bar{s}_{mj}^{1}}{\hat{s}_{mj}^{1}}$. Thus, we can also write $\tilde{s}_{mj}^{1} = \bar{s}_{mj}^{1} \pm \hat{s}_{mj}^{1} a_{mj}$. Furthermore, $\tilde{s}_{mj}$ and $\tilde{s}_{mp}$ take values in the intervals $[\bar{s}_{mj} - \hat{s}_{mj}, \bar{s}_{mj} + \hat{s}_{mj}]$ and $[\bar{s}_{mp} - \hat{s}_{mp}, \bar{s}_{mp} + \hat{s}_{mp}]$, respectively. We consider the scale deviation of $\tilde{s}_{mj}$ and $\tilde{s}_{mp}$ from their nominal values as $b_{mj} = \frac{\tilde{s}_{mj} - \bar{s}_{mj}}{\hat{s}_{mj}}$ and $c_{mp} = \frac{\tilde{s}_{mp} - \bar{s}_{mp}}{\hat{s}_{mp}}$ that belongs to [−1, 1]. Similarly, the random treatment times following the first one for patients on the waiting list and treatment times for booked patients might be rewritten as $\tilde{s}_{mj} = \bar{s}_{mj} \pm b_{mj}\hat{s}_{mj}$ and $\tilde{s}_{mp} = \bar{s}_{mp} \pm c_{mp}\hat{s}_{mp}$, respectively. In what follows, we provide the robust counterpart of RAS model (27)-(48) according to the above-mentioned uncertain intervals.

4.2.1. Robust counterpart of RAS problem

Constraints (37) in the deterministic model (27) - (48) are affected by uncertain treatment times. These constraints indicate that the total time needed to treat patients on each machine in each shift, on a day must not exceed the machine capacity. Recall from chapter 3 that the schedule is robust if constraint (37) is satisfied for the worst-case treatment times under a given budget of uncertainty. In other words, robust counterpart of (37) is equivalent to satisfying this constraint in case the total treatment time of all patients (left hand side of (37)) takes its maximum value. This is equivalent to the case where the uncertain treatment times (i.e. $\tilde{s}_{mj}$, $\tilde{s}_{mp}$, and $\tilde{s}_{mj}^{1}$) take their maximum value under a given budget of uncertainty. Accordingly, for a given *m, k,* and *f* we define a budget of uncertainty, $\Gamma_{mkf}^{1}, \Gamma_{mkf}^{2}$ and Γ_{mkf}^{3} for $\tilde{s}_{mj}, \tilde{s}_{mp}, \tilde{s}_{mj}^{1}$, that indicate the maximum number of

uncertain treatment times that can take their worst-case value. Let z^*_{mjkf}, y^*_{mjkf} and $\overline{y}^*_{pkf}$ be the optimal schedules of booked and new patients, the following protection function is defined for (37).

$$Max \sum_{j=1}^{|WP|}(\hat{s}^1_{mj}\, a_{mj}\, z^*_{mjkf} + \hat{s}_{mj}\, b_{mj}\, y^*_{mjkf}) + \sum_{p=1}^{|BP|} \hat{s}_{mp}\, c_{mp}\, \overline{y}^*_{pkf}\, av_{pkf} \quad (50)$$

$s.t.$:

$$\sum_{j=1}^{|WP|} a_{mj} \le \Gamma^1_{mkf} \qquad \forall m \in M, k \in K, f \in F \ (51)$$

$$\sum_{j=1}^{|WP|} b_{mj} \le \Gamma^2_{mkf} \qquad \forall m \in M, k \in K, f \in F \ (52)$$

$$\sum_{p=1}^{|BP|} c_{mp} \le \Gamma^3_{mkf} \qquad \forall m \in M, k \in K, f \in F \ (53)$$

$$0 \le a_{mj} \le 1 \qquad \forall m \in M, j \in WP \ (54)$$

$$0 \le b_{mj} \le 1 \qquad \forall m \in M, j \in WP \ (55)$$

$$0 \le c_{mp} \le 1 \qquad \forall m \in M, p \in BP \ (56)$$

For the above protection function, we define $e_{mkf}, g_{mj}, h_{mkf}, i_{mj}, l_{mkf}$ and n_{mp} as the dual variables corresponding to its constraints. The dual counterpart of this protection function is the following optimization problem.

$$Min\ \Gamma^1_{mkf}\, e_{mkf} + \Gamma^2_{mkf}\, h_{mkf} + \Gamma^3_{mkf}\, l_{mkf} + \sum_{j=1}^{|WP|}(g_{mj} + i_{mj}) + \sum_{p=1}^{|BP|} n_{mp} \quad (57)$$

s.t.:

$$e_{mkf} + g_{mj} \ge \hat{s}^1_{mj}\, z_{mjkf} \qquad \forall m \in M, j \in WP, k \in K, f \in F \ (58)$$

$$h_{mkf} + i_{mj} \ge \hat{s}_{mj}\, y_{mjkf} \qquad \forall m \in M, j \in WP, k \in K, f \in F \ (59)$$

$$l_{mkf} + n_{mp} \ge \hat{s}_{mp}\, av_{pkf}\, \overline{y}_{pkf} \qquad \forall m \in M, p \in BP, k \in K, f \in F \ (60)$$

$$e_{mkf}, h_{mkf}, l_{mkf} \ge 0 \qquad \forall m \in M, k \in K, f \in F \ (61)$$

$$g_{mj}, i_{mj}, n_{mp} \ge 0 \qquad \forall m \in M, p \in BP \ (62)$$

We finally substitute the dual of the protection function in constraint (37) in order to find its robust counterpart as follows:

$$\sum_{j=1}^{|WP|}(\bar{s}_{mj}^{1} Z_{mjkf}+\bar{s}_{mj}y_{mjkf}) + \sum_{p=1}^{|BP|}\bar{s}_{mp}av_{pkf}\bar{y}_{pkf} + \Gamma_{mkf}^{1} e_{mkf} + \Gamma_{mkf}^{2} h_{mkf} + \Gamma_{mkf}^{3} l_{mkf} + \sum_{j=1}^{|WP|}(g_{mj} + i_{mj}) + \sum_{p=1}^{|BP|} n_{mp} \leq T_{mkf} \quad \forall m \in M, k \in K, f \in F \quad (63)$$

$$e_{mkf} + g_{mj} \geq \hat{s}_{mj}^{1} z_{mjkf} \quad \forall m \in M, j \in WP, k \in K, f \in F \quad (64)$$

$$h_{mkf} + i_{mj} \geq \hat{s}_{mj} y_{mjkf} \quad \forall m \in M, j \in WP, k \in K, f \in F \quad (65)$$

$$l_{mkf} + n_{mp} \geq \hat{s}_{mp} av_{pkf}\overline{y}_{pkf} \quad \forall m \in M, p \in BP, k \in K, f \in F \quad (66)$$

$$e_{mkf}, h_{mkf}, l_{mkf} \geq 0 \quad \forall m \in M, k \in K, f \in F \quad (67)$$

$$g_{mj}, i_{mj}, n_{mp} \geq 0 \quad \forall m \in M, p \in BP, j \in WP \quad (68)$$

Finally, the robust counterpart of the deterministic model can be formulated as follows:

$$\text{Max} \sum_{j=1}^{|WP|}\sum_{m=1}^{|M|} w_j X_{mj} \quad (69)$$

Subject to

(28) - (36)

$$\sum_{j=1}^{|WP|}(\bar{s}_{mj}^{1} Z_{mjkf}+\bar{s}_{mj}y_{mjkf}) + \sum_{p=1}^{|BP|}\bar{s}_{mp}av_{pkf}\bar{y}_{pkf} + \Gamma_{mkf}^{1} e_{mkf} + \Gamma_{mkf}^{2} h_{mkf} + \Gamma_{mkf}^{3} l_{mkf} + \sum_{j=1}^{|WP|}(g_{mj} + i_{mj}) + \sum_{p=1}^{|BP|} n_{mp} \leq T_{mkf} \quad \forall m \in M, k \in K, f \in F \quad (70)$$

$$e_{mkf} + g_{mj} \geq \hat{s}_{mj}^{1} z_{mjkf} \quad \forall m \in M, j \in WP, k \in K, f \in F \quad (71)$$

$$h_{mkf} + i_{mj} \geq \hat{s}_{mj} y_{mjkf} \quad \forall m \in M, j \in WP, k \in K, f \in F \quad (72)$$

$$l_{mkf} + n_{mp} \geq \hat{s}_{mp} av_{pkf}\overline{y}_{pkf} \quad \forall m \in M, p \in BP, k \in K, f \in F \quad (73)$$

(36) - (48)

$$e_{mkf}, h_{mkf}, l_{mkf} \geq 0 \quad \forall m \in M, k \in K, f \in F \quad (74)$$

$$g_{mj}, i_{mj}, n_{mp} \geq 0 \quad \forall m \in M, p \in P, j \in WP \quad (75)$$

The robust counterpart of model (27)-(48) is a mixed-integer program that contains more decision variables and constraints compared to the deterministic model. Nevertheless, it has a similar structure as deterministic model; hence, similar solution algorithms can be employed to efficiently solve this model. Also, this model protects patients' schedules against the uncertainty in treatment times. In other words, under a given budget of uncertainty, the feasibility of schedules with regard to the capacity of linacs during regular shifts is guaranteed in the robust RAS model.

Chapter 5: Implementation of metaheuristics on radiotherapy appointment scheduling model

In this chapter we explain how the metaheuristics discussed in chapter 3 are implemented on our models by the aid of a small example. Recall from chapter 3, all metaheuristics start with a population of solutions. In what follows, we first describe how these populations are generated; then we confine our attention to the details of operations used in each algorithm.

5.1. Population generation details

Recall from chapter 4, our binary integer programming model consists of 5 decision variables, $\bar{y}_{pkf}, \bar{Z}_{pkf}, Z_{mjkf}, y_{mjkf}$, and X_{mj}. Thus, an index permutation matrix including 8 rows is defined as a representative of a feasible solution (schedule). The first 3 rows of this matrix are dedicated to the indices of variables $\bar{y}_{pkf}$ and $\bar{Z}_{pkf}$; the next 4 rows are dedicated to the indices of variables Z_{mjkf} and y_{mjkf}; and the last row is designated for index j of decision variable X_{mj}. Each row defines an order for the indices, so that the combination of rows determines the permutation of the decision variable indices, by which binary variables are assigned. For instance, assume that we have a system in which the booked patients belong to p = 1,2; the machines belong to m =1, 2; the patients on the waiting list belong to j = 1,2,3; the weekdays in time horizon belong to k = 1,2,3,4,5,6; and the shifts in each day belong to f = 1,2. Thus, the dimension of the matrix is 8×6, in which 6 is the maximum of all the indices, and the entries in the matrix are random numbers uniformly distributed in [0, 1]. Table (4) provides an example for the permutation matrix.

Table 4- index permutation matrix

index	1	2	3	4	5	6
p	0.78303	0.78788				
k	0.73586	0.75928	0.82658	0.5581	0.85493	0.35336
f	0.01406	0.43094				
m	0.69724	0.68427				
j	0.01628	0.36028	0.44087			
k	0.45438	0.36547	0.39722	0.87565	0.77292	0.93509
f	0.56892	0.03699				
j	0.9277	0.56323	0.70615			

In each row of table (4), the random numbers in [0, 1] are only assigned to cells belonging to the range of each index. The rest of cells are assigned an infinite value, which means they cannot take any value. For instance, p =1, 2; hence, the first two elements of row p are assigned random numbers and the rest are infinity. After generating the aforementioned random numbers, the elements of each row (except the last row) are sorted and priorities are assigned to indices in such a way that the index with the highest random value has the lowest priority and the one with the smallest value has the highest priority as depicted in table (5).

Table 5- prioritized indices in the index permutation matrix

index	1	2	3	4	5	6
p	1	2				
k	3	4	5	2	6	1
f	1	2				
m	2	1				
j	1	2	3			
k	4	6	5	2	3	1
f	1	2				

According to table (5), the combination of the first 3 rows gives us the permutations of *p, k*, and *f* indices for variables $\overline{y}_{pkf}$ and $\bar{Z}_{pkf}$, and the combination of next 4 rows determines the permutations of *m, j, k, f* indices for variables Z_{mjkf} and y_{mjkf}. Therefore, the permutations for *(p, k, f)* respectively are: (1,6,1)- (1,6,2)- (1,4,1)- (1,4,2)- (1,1,1)- (1,1,2)- (1,2,1)- (1,2,2)- (1,3,1)- (1,3,2)- (1,5,1)- (1,5,2)- (2,6,1)- (2,6,2)- (2,4,1)- (2,4,2)- (2,1,1)- (2,1,2)- (2,2,1)- (2,2,2)- (2,3,1)- (2,3,2)- (2,5,1)- (2,5,2). In a similar fashion, the order of permutations for *m, j, k, f* indices of variables Z_{mjkf} and y_{mjkf}could be obtained. According to this order of permutations, the variables are plugged into the *fitness* function (see *Algorithm 5*) and binary values are assigned to them accordingly. Therefore, patient 1 in day 6 and shift 1 is checked first; if all the constraints in the fitness function are satisfied, $\overline{y}_{161}$ and $\bar{Z}_{161}$ take 1; otherwise, they remain as 0. This is followed by patient 1, day 6 and shift 2, and so on. It is worth noting that the constraints indicated in *Algorithm 5* are described in chapter 4 (section 4.1.2 – model (27) - (48)).

Algorithm 5- Fitness function for RAS model

1. Get the index permutation matrix;
2. For all permutations of *(p, k, f)* in the obtained order in the index permutation matrix
   2.1. If all constraints (44), (38), (43), (41), (37) (in the mentioned sequence) are satisfied
      2.1.1. Assign 1 to decision variable $\overline{y}_{pkf}$: $\overline{y}_{pkf}$
   2.2. If all constraints (45), (40), (39), (40) (in the mentioned sequence) are satisfied
      2.2.1. Assign 1 to decision variable $\overline{Z}_{pkf}$:$\bar{Z}_{pkf} = 1$;
3. For all permutations of *(m, j, k, f)* in the obtained order in the index permutation matrix
   3.1. If all constraints (29), (28), (30), (32), (35), (36), (37) (in the mentioned sequence) are satisfied
      3.1.1. Assign 1 to decision variable Z_{mjkf}: Z_{mjkf};
   3.2. Assign binary values to X_{mj} according to the last row of the index permutation matrix;
   3.3. Check constraint (28) and revise the value assign to X_{mj} in case of infeasibility;
   3.4. If all constraints (31), (35), (33), (31), (32), (37), (34) (in the mentioned sequence) are satisfied
      3.4.1. Assign 1 to decision variable y_{mjkf}: y_{mjkf};
4. Calculate the objective function and return it.

The last row of permutation matrix is associated with assigning machines to each patient on the waiting list (decision variable X_{mj}). To this end, we multiply this row by 2 and subtract by 1 to create numbers in [-1, 1]. Then, the values less than zero (in [-1, 0]) are assigned 0; in other words, this space could be exploited to determine patients that are not assigned to any machine; the positive values remain with no change. The new row is multiplied by the number of machines which is 2, and rounded up as depicted in table (6). In this way, patients who are supposed to be assigned to a machine (in [0, 1]), are assigned to one of machines 1 and 2.

Table 6- permutation matrix for machine assignment to patients

index	1	2	3	4	5	6
j	0.9277	0.56323	0.70615			
( )×2-1	0.85539	0.12646	0.41231			
×m	1.71078	0.25292	0.82462			
round up	2	1	1			

According to the last row, the first patient has been assigned to machine 2, and the rests are assigned to machine 1. These assignments are checked again by *fitness* function (*Algorithm 5*) and corrected in case of infeasibility. Finally, the *fitness* function returns the solution obtained and also the objective value to the algorithms.

It is worth mentioning here that the *fitness* function is coded in MATLAB and provided in Appendix (section 9.1). In what follows, we explain the procedures performed for each of the metaheuristic algorithms described in section 3.

5.2. Whale Optimization Algorithm

This section provides the operations used in WOA. Let's consider a population of 2 solutions (schedules). Thus, 2 index permutation matrices have been randomly generated as depicted in tables (7) and (8). Recall from section 5.1, these matrices correspond to two different schedules and are generated randomly for two machines, two booked patients, 6 days, two shifts, and 3 patients on the wating list. The patients' priority is assumed to decrease from 3 to 1.

Table 7- initial index permutation matrix (a) for WOA

index	1	2	3	4	5	6
p	0.4636	0.48065				
k	0.84304	0.93951	0.26635	0.03943	0.3367	0.09388
f	0.28527	0.10426				
m	0.49187	0.72472				
j	0.38461	0.48464	0.96331			
k	0.93221	0.97896	0.01738	0.050759	0.88065	0.04097
f	0.34743	0.83789				
j	0.07515	0.76332	0.94705			

Table 8- initial index permutation matrix (b) for WOA

index	1	2	3	4	5	6
p	0.12656	0.51339				
k	0.39278	0.3288	0.24844	0.02611	0.83368	0.6318
f	0.28668	0.02227				
m	0.59807	0.01528				
j	0.86327	0.24522	0.60927			
k	0.35519	0.16909	0.73519	0.96691	0.83503	0.83197
f	0.51423	0.6731				
j	0.69212	0.95076	0.47988			

By using *fitness* function, the 2 schedules (only X_{mj} decision variable is presented) would be as shown in tables (9) and (10):

Table 9- the schedule corresponding to index permutation matrix (a)

X_{mj}			Obj.
0	0	0	3
0	1	1	

Table 10- the schedule corresponding to index permutation matrix (b)

X_{mj}			Obj.
1	0	0	5
0	1	0	

Then for the maximum number of iterations, we follow the procedure below:

First, parameters *a, A, C, l* and *p* are updated. Recall from chapter 3, a decreases from 2 to 0, its value can be updated as $a = 2 - current\ iteration\ number. \frac{2}{maximum\ number\ of\ iterations}$. Also, *r* is random vector in [0, 1]; while *A* and *C* are updated according to equations (10) and (11) (chapter 3), respectively. As an example, assume that at an iteration, the algorithm has updated parameters values as depicted in table (11):

Table 11- updated parameters for shrinking mechanism in the exploration phase(WOA)

p	0.23908
a	1.91931
r	0.12107
A	-1.4546
C	0.24215

According to the above values, since p < 0.5 and $|A|>1$, the algorithm chooses the exploration phase with shrinking mechanism. Thus, the schedule of current population is updated by equations (8) and (9) based on a randomly chosen schedule.

As an example, we assume that the index permutation matrix (b) (table (8)) is the randomly chosen one as depicted in table (12).

Table 12- randomly selected matrix from initial population (WOA)

index	1	2	3	4	5	6
p	0.12656	0.51339				
k	0.39278	0.3288	0.24844	0.02611	0.83368	0.6318
f	0.28668	0.02227				
m	0.59807	0.01528				
j	0.86327	0.24522	0.60927			
k	0.35519	0.16909	0.73519	0.96691	0.83503	0.83197
f	0.51423	0.6731				
j	0.69212	0.95076	0.47988			

Then each schedule matrix ((a) and (b)) is updated using equations (8) and (9). Table 13 provides vector D in exploration phase with shrinking mechanism. Therefore, the updated index permutation matrices for the members of the population would be tables (14) and (16).

Table 13- updated vector D for matrix (a) in exploration phase with shrinking mechanism (WOA)

index	1	2	3	4	5	6
p	0.43295	0.35633				
k	0.47793	0.85989	0.20619	0.03311	0.13483	0.05911
f	0.21585	0.09887				
m	0.34705	0.72102				
j	0.17558	0.42526	0.81577			
k	0.8462	0.93801	0.16063	0.27346	0.67845	0.05911
f	0.22292	0.6749				
j	0.09245	0.5331	0.83084			

Table 14- updated position for matrix (a) in exploration phase with shrinking mechanism (WOA)

index	1	2	3	4	5	6
p	0.75631	1.0317				
k	1.48069	1.57957	0.5483	0.07426	1.0298	0.71778
f	0.60065	0.16608				
m	1.10288	1.06405				
j	1.11865	0.86378	1.79586			
k	1.58605	1.53349	0.96884	1.36467	1.82188	1.06541
f	0.83847	1.65478				
j	0.82659	1.72618	1.68839			

Now, the WOA algorithm fits the matrix with search space boundaries ([0, 1]) (i.e., the whole row is divided by the ceil of maximum value at the row) and assigns binary values according to the *fitness* function (*Algorithm 5*). Table (15) presents the updated schedule (a) (only X_{mj} is provided).

Table 15- updated schedule corresponding to matrix (a) in exploration phase with shrinking mechanism

X_{mj}			Obj.
1	0	0	5
0	1	0	

Finally, for the second index permutation matrix (b), the updated position is presented in table (16).

Table 16- updated vector D for matrix (b) in exploration phase with shrinking mechanism (WOA)

index	1	2	3	4	5	6
p	0.27528	0.10155				
k	0.79696	0.77733	0.95938	0.33504	0.66054	0.22414
f	0.06874	0.30799				
m	0.05219	0.85847				
j	0.92193	0.76524	0.45982			
k	0.28248	0.03612	0.06478	0.2514	0.52062	0.70711
f	0.14794	0.34212				
j	0.35238	0.12357	0.9122			

Table 17- updated position for matrix (b) in exploration phase with shrinking mechanism (WOA)

index	1	2	3	4	5	6
p	0.26608	1.07932				
k	0.96409	0.80705	0.60409	0.06409	2.04632	1.5508
f	0.70368	0.05466				
m	1.46801	0.03751				
j	2.11849	0.60191	1.49548			
k	0.87183	0.41504	1.80456	2.37334	2.04964	2.04213
f	1.26221	1.65217				
j	1.69886	2.3337	1.1779			

Fitting with boundaries and constraints, we have the new schedule (b) as depicted in table (18).

Table 18- updated schedule corresponding to matrix (b) in exploration phase with shrinking mechanism

X_{mj}			Obj.
1	0	0	3
0	0	0	

Afterwards, assume that in another iteration of the WOA, the updated parameters are those provided in table (19).

Table 19- updated parameters for spiral-shaped mechanism (WOA)

p	0.53193
a	1.53008
r	0.40884
A	-0.279
C	0.81768

Because $p > 0.5$, the algorithm chooses the spiral-shaped mechanism and update the matrices using equation (14). In this example the best matrix so far is schedule presented in table (14) and current positions of matrices in the population are provided in tables (14) and (16). Also, Recall from chapter 3, l is generated as a random vector in [-1, 1] (e.g., $l = 0.12146$).

Using equation (14), for the schedule (a) (the matrix depicted in table (14) we have the updated vector *D',* updated position, and the new schedule for the first member of population in tables (20) to (22).

Table 20- updated vector D' for schedule (a) in spiral-shaped mechanism

index	1	2	3	4	5	6
p	0	0				
k	0	0	0	0	0	0
f	0	0				
m	0	0				
j	0	0	0			
k	0	0	0	0	0	0
f	0	0				
j	0	0	0			

Table 21-updated position for matrix (a) in spiral-shaped mechanism

Index	1	2	3	4	5	6
p	0.75631	0.0317				
k	0.48068	0.57957	0.54835	0.07426	1.0298	0.71778
f	0.60064	0.16608				
m	0.10288	0.06405				
j	0.11865	0.86378	0.79586			
k	0.58604	0.53349	0.96884	0.36467	0.82188	0.06541
f	0.83847	0.65478				
j	0.82658	0.72618	0.68839			

Table 22- updated schedule corresponding to matrix (a) in spiral-shaped mechanism

X_{mj}			Obj.
0	1	1	6
1	0	0	

And finally, the updated vector *D',* updated position for the schedule (b) (table 16), and the new schedule for the second member of population are depicted in tables (23) to (25).

Table 23- updated vector D' for schedule (b) in spiral-shaped mechanism

Index	1	2	3	4	5	6
p	0.49023	1.04762				
k	0.48341	0.22749	0.06145	0.01017	1.01652	0.83302
f	0.10303	0.11142				
m	1.36512	0.02655				
j	2.00028	0.26188	0.69961			
k	0.28579	0.11845	0.83572	2.00867	1.22776	1.97672
f	0.26064	0.78391				
j	0.12773	0.39248	0.5105			

Table 24- updated position for matrix (b) in spiral-shaped mechanism

Index	1	2	3	4	5	6
p	0.81962	2.26223				
k	1.50992	1.06362	0.67918	0.07558	3.19412	2.49139
f	0.82002	0.18047				
m	3.00942	0.06748				
j	4.37754	0.8976	2.28544			
k	1.19453	0.54878	2.7482	4.6414	3.43595	4.27412
f	1.39342	2.32383				
j	0.84308	0.77687	0.75432			

Table 25- updated schedule corresponding to matrix (b) in spiral-shaped mechanism

X_{mj}			Obj.
0	0	0	6
1	1	1	

As an example of an exploitation phase in shrinking mechanism in the algorithm, assume that the updated parameter values are as depicted in table (26).

Table 26- updated parameters for exploitation phase in shrinking mechanism

p	0.03892
a	0.63477
r	0.20229
A	-0.378
C	0.40459

The best index permutation matrix so far is the one shown in table (21). Current positions of the matrices in the population are depicted in tables (21) and (24). Using equations (12) and (13), for the first schedule (index permutation matrix (a) (table (21)) we have the updated vector *D*, updated position, and the new schedule for the first member of population in tables (27) to (29).

Table 27- updated vector D for schedule (a) in exploitation phase with shrinking mechanism

Index	1	2	3	4	5	6
p	0.0.45031	0.01887				
k	0.28620	0.34508	0.32649	0.04422	0.61316	0.42738
f	0.35763	0.34508				
m	0.06125	0.03814				
j	0.07064	0.51431	0.47386			
k	0.34893	0.31764	0.57686	0.21713	048936	003895
f	0.49923	0.38987				
j	0.49216	0.43238	0.40987			

Table 28- updated position for the matrix (a) in exploitation phase with shrinking mechanism

Index	1	2	3	4	5	6
p	0.92651	0.03883				
k	0.58886	0.70999	0.67175	0..9098	1.26154	0.87931
f	0.73581	0.20345				
m	0.12603	007847				
j	0.14535	1.05817	0.97496			
k	0.71792	0.65354	1.18687	0.44673	1.00683	0.8013
f	1.02716	0.80213				
j	1.01260	0.8896	0.84330			

Table 29- updated schedule corresponding to matrix (a) in exploitation phase with shrinking mechanism

X_{mj}			Obj.
0	0	0	3
0	1	1	

Finally, we have similar results for the schedule (b) (table (24)) in tables (30) to (32).

Table 30- updated vector D for schedule (b) in exploitation phase with shrinking mechanism

index	1	2	3	4	5	6
p	0.51363	2.2494				
k	1.31544	0.82943	0.45732	0.04553	2.77747	2.20099
f	0.57701	0.11328				
m	2.9678	0.04157				
j	4.32953	0.54813	1.96345			
k	0.95742	0.33294	2.35622	4.49386	3.10342	4.24766
f	1.05418	2.05891				
j	0.50866	0.48306	0.4758			

Table 31- updated position for matrix (b) in exploitation phase with shrinking mechanism

index	1	2	3	4	5	6
p	0.95044	0.88186				
k	0.97786	0.89305	0.7212	0.09147	2.07955	1.54965
f	0.81873	0.20889				
m	1.22457	0.07976				
j	1.75501	1.07095	1.53795			
k	0.94791	0.65932	1.85939	2.06313	1.99482	1.67082
f	1.2369	1.43295				
j	1.01884	0.90876	0.86623			

Table 32- updated schedule corresponding to matrix (b) in exploitation phase with shrinking mechanism

X_{mj}			Obj.
0	0	0	3
0	1	1	

WOA is coded in MATLAB (see Appendix in 9.2).

5.3. Particle Swarm Optimization

This section provides the operations used in PSO. Let's consider a population of 2 solutions (schedules). Thus, 2 index permutation matrices have been randomly generated as depicted in tables (33) and (34). Recall from section 5.1, these matrices correspond to two different schedules and are generated randomly for two machines, two booked patients, 6 days, two shifts, and 3 patients on the wating list. The patients' priority is assumed to decrease from 3 to 1.

Table 33- initial index permutation matrix (a) for PSO

index	1	2	3	4	5	6
p	0.0502	0.59834				
k	0.90856	0.07201	0.02827	0.54202	0.2774	0.18616
f	0.3544	0.42172				
m	0.13834	0.56272				
j	0.18094	0.31042	0.75462			
k	0.97621	0.73559	0.73529	0.77261	0.81971	0.62075
f	0.77979	0.76131				
j	0.40976	0.23814	0.64819			

Table 34- initial index permutation matrix (b) for PSO

index	1	2	3	4	5	6
p	0.72293	0.06117				
k	0.14396	0.78338	0.76538	0.64839	0.39877	0.90142
f	0.27883	0.62092				
m	0.48864	0.82401				
j	0.67109	0.65851	0.27301			
k	0.38742	0.31073	0.33147	0.24409	0.75742	0.92352
f	0.16393	0.23124				
j	0.06473	0.6748	0.46191			

By using *fitness* function the 2 schedules (only X_{mj} decision variable is presented) would be as shown in tables (35) and (36):

Table 35- the schedule corresponding to index permutation matrix (a)

X_{mj}			Obj.
0	0	1	1
0	0	0	

Table 36- the schedule corresponding to index permutation matrix (b)

X_{mj}			Obj.
0	1	0	2
0	0	0	

So far, tables (33) and (34) are the particle's best known positions and table (35) is the global best known position since it has the highest objective value. The velocity has been initialized as 0. Recall from section 3.2.2, ω is velocity importance coefficient; φ_p is the personal best learning coefficient; φ_g is the global best learning coefficient; and r_p and r_g are randomly generated parameters. These parameters are set as table (37).

Table 37- parameters for PSO

ω	1
φ_p	2
φ_g	2
r_p	0.64228
r_g	0.35753

Therefore, the particle's velocity is updated using equation (15) as depicted in tables (38) and (39):

Table 38- updated velocity for the matrix (a)

index	1	2	3	4	5	6
p	0.48105	-0.3841				
k	-0.5467	0.50867	0.52709	-0.1024	0.26528	0.51146
f	-0.054	0.14244				
m	0.25049	0.18648				
j	0.35049	0.24891	-0.3444			
k	-0.421	-0.3038	-0.2888	-0.3779	-0.0445	0.2165
f	-0.4404	-0.379				
j	-0.2467	0.31224	-0.1332			

Table 39- updated velocity for matrix (b)

index	1	2	3	4	5	6
p	0	0				
k	0	0	0	0	0	0
f	0	0				
m	0	0				
j	0	0	0			
k	0	0	0	0	0	0
f	0	0				
j	0	0	0			

Then the particles' positions are updated using equation (16) and fitted with *fitness* function as depicted in tables (40) - (43).

Table 40- updated position for index permutation matrix (a)

index	1	2	3	4	5	6
p	0.53125	0.21423				
k	0.36182	0.58069	0.55536	0.43959	0.54269	0.69762
f	0.30036	0.56416				
m	0.38883	0.74956				
j	0.53143	0.55933	0.41023			
k	0.55519	0.43178	0.44653	0.39468	0.77517	0.83725
f	0.3394	0.38227				
j	0.16304	0.55039	0.51498			

Table 41- updated schedule corresponding to index permutation matrix (a)

X_{mj}			Obj.
0	1	1	3
0	0	0	

Table 42- updated position for index permutation matrix (b)

index	1	2	3	4	5	6
p	0.72293	0.06117				
k	0.14396	0.78338	0.76538	0.39877	0.64839	0.90142
f	0.27883	0.62092				
m	0.48864	0.82401				
j	0.67109	0.65851	0.27301			
k	0.38742	0.31073	0.33147	0.24409	0.75742	0.92352
f	0.16393	0.23124				
j	0.06473	0.6748	0.46191			

Table 43- updated schedule corresponding to matrix (b)

X_{mj}			Obj.
0	1	0	2
0	0	0	

Now the schedule in table (40) with the objective value of 3 is the global best known particle. These operations continue until we reach the maximum number of iterations. PSO is coded in MATLAB (see Appendix in 9.3).

5.4. Firefly Algorithm

This section provides a small example on the implementation details of the Firefly algorithm. Let's consider the population of 3 fireflies (index permutation matrices). The initial population is generated randomly and fitted by *fitness* function as depicted in tables (44) – (49):

Table 44- initial index permutation matrix (a) for FA

index	1	2	3	4	5	6
p	0.10523	0.57438				
k	0.19983	0.29994	0.94442	0.22135	0.8185	0.07545
f	00.90172	0.06238				
m	0.86607	0.46082				
j	0.69062	0.83844	0.32285			
k	0.33835	0.68923	0.19952	0.71903	0.57006	0.64212
f	0.9651	0.19841				
j	0.81869	0.73209	0.17082			

Table 45- schedule for corresponding to matrix (a)

X_{mj}			Obj.
0	1	0	5
1	0	0	

Table 46- initial index permutation matrix (b) for FA

index	1	2	3	4	5	6
p	0.60899	0.75672				
k	0.85433	0.56268	0.81677	0.30833	0.89148	0.93103
f	0.7327	0.3041				
m	0.27032	0.92639				
j	0.75749	0.76524	0.66652			
k	0.66608	0.89877	0.70059	0.80023	0.41961	0.89371
f	0.14692	0.21798				
j	0.57543	0.83622	0.7421			

Table 47- schedule corresponding to matrix (b)

X_{mj}			Obj.
1	0	1	6
0	1	0	

Table 48- initial index permutation matrix (c) for FA

index	1	2	3	4	5	6
p	0.49681	0.48742				
k	0.69213	0.82618	0.0403	0.53828	0.01229	0.50142
f	0.08449	0.5544				
m	0.99588	0.68717				
j	0.11088	0.30472	0.09906			
k	0.74118	0.26566	0.27693	0.73223	0.15178	0.44415
f	0.38212	0.31457				
j	0.09647	0.24814	0.58953			

Table 49- schedule corresponding to matrix (c)

X_{mj}			Obj.
0	0	1	1
0	0	0	

According to tables (44) – (49), the first firefly (table (44)), with light intensity (objective value) of 5, searches for a firefly (schedule) with higher light intensity and moves toward it. Thus, the first firefly (table (44)) would move toward second one (table (46)) and will have one new position. The second firefly (table (46)) with the light intensity of 6 does not move toward any firefly, since it has the highest light intensity. Finally, the third firefly (table (48)) with the light of 1 moves toward the first and the second ones with higher light intensity. As a result, we will have three new positions in addition to the previous three positions we had. These 6 schedules are sorted according to their light intensity (objective values) and the three positions with the highest light intensity would be accepted as the new population for the next iteration. This procedure will continue until we reach the maximum number of iterations. Finally, the firefly with the highest objective value (light intensity) is chosen as the best schedule. Tables (51) to (53) provide the movement operations of firefly 1 (table (44)) toward firefly 2 (table (46)). The parameters are initialized as shown in table (50).

Table 50- Parameters for FA

β	0.5
γ	1
α	0.9

ϵ_t is a vector drawn from a uniform distribution in [0, 1] (see table (51)).

Table 51- uniformly distributed vector ϵ_t in [0, 1]

index	1	2	3	4	5	6
p	0.26277	0.31005				
k	0.52464	0.46388	0.29767	0.55262	0.90479	0.59898
f	0.19067	0.52165				
m	0.95096	0.79364				
j	0.38914	0.17041	0.66954			
k	0.06969	0.7527	0.19832	0.81755	0.12988	0.74237
f	0.46773	0.3025				
j	0.77929	0.32806	0.75621			

Equation (25) provides the new position of the index permutation matrix (a) (table (44)) as:

Table 52- updated position of index permutation matrix (a)

index	1	2	3	4	5	6
p	0.53715	0.94161				
k	0.88522	0.84004	1.14953	0.76187	1.66911	0.82027
f	0.99119	0.64587				
m	1.51305	1.36252				
j	1.07414	0.9554	1.07813			
k	0.54825	1.46699	0.57292	1.49516	0.61341	1.42833
f	1.1766	0.48044				
j	1.40541	1.07885	1.05751			

Table 53- updated schedule corresponding to matrix (a)

X_{mj}			Obj.
1	1	1	6
0	0	0	

FA is coded in MATLAB (Appendix section 9.4).

Chapter 6: Experimental result

In this chapter we present some numerical results on the application of proposed robust RAS model as well as the 3 metaheuristics on a real case study adopted from literature (Conforti et al. 2010). Our goal is twofold: *i)* to analyze the relationship between the budget of uncertainty and the feasibility/cost of robust appointment schedules; and *ii)* to compare the performance of 3 metaheuristics with a commercial solver in terms of solution quality and CPU time.

6.1. Case data and implementation details

Adopted from (Conforti et al. 2010), we consider two test instances corresponding to a radiotherapy clinic characterized by the following primary data:

- Two machines (linear accelerators) M_1 and M_2 (test instance 1) and three machines (test instance 2)
- planning horizon of 6 days (from Monday to Saturday);
- Two shifts per day.
- $|BP| = 76$ already booked patients. The number of treatment sessions to be carried out is equal to 4 or 5 for all patients. The treatment time is patient-dependent and lasts either 10 or 15 minutes (used as nominal values). For each patient p, the last treatment day of the previous week is known (lpt_p). This information is related to the maximum feasible number of days between two consecutive weekly sessions and is required to determine the first treatment session in the current week.
- $|WP| = 40$ patients waiting to start their weekly treatment. The number of treatment sessions (t) to be carried out for them during the planning horizon, is set to 5 for all patients on this list. The treatment times are patient-dependent; in addition, it is assumed that they are the same on both machines. Patients are grouped according to the assigned priority. *pr=10* is assigned if the patient belongs to priority class *B*; whereas *pr=1* is assigned to a patient who belongs to priority class *C*.
- $T_{mkf} = 300; \forall\, m, k, f.$
- Each booked patient is available either in the morning or in the afternoonon each day;
- $delay_p = 1\ \forall p \in BP.$

parameters related to patients on the waiting list are summarized in Table (54).

Table 54- Nominal treatment times and assigned priority values for patients on WP list

patient on *WP*	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
set^1	20	10	10	10	15	15	15	15	15	15	15	15	10	10	10	10	10	10	10	15
set	8	5	5	5	7	7	7	7	7	7	7	7	5	5	5	5	5	5	7	5
s	4	2	2	2	3	3	3	3	3	3	3	3	4	4	4	4	3	3	3	3
$\bar{s}^1$	24	12	12	12	18	18	18	18	18	18	18	18	14	14	14	14	13	13	13	18
$\bar{s}$	12	7	7	7	10	10	10	10	10	10	10	10	9	9	9	9	8	8	10	8
pr	10	1	10	10	10	10	10	10	10	1	1	1	1	10	10	10	10	10	10	10
patients on *WP*	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
set^1	15	15	15	15	15	15	10	10	10	10	10	10	10	10	10	10	10	10	10	10
set	5	5	5	5	5	5	7	7	7	7	7	7	7	7	7	5	5	5	5	5
s	3	3	3	3	3	3	4	4	4	2	2	2	2	2	2	2	2	2	2	2
$\bar{s}^1$	18	18	18	18	18	18	14	14	14	12	12	12	12	12	12	12	12	12	12	12
$\bar{s}$	8	8	8	8	8	8	11	11	11	9	9	9	9	9	9	7	7	7	7	7
pr	10	10	10	10	1	1	1	1	1	1	1	1	1	1	1	1	10	10	10	10

Also, the priority weights in the objective function of RAS model are determined by seting *a=1 and b=3.*

Both the deterministic and robust RAS models are solved by using the *MIP* solver of *CPLEX 12.6.1*, on a PC Pentium IV, under the Windows 7 Enterprise operating system , with 2 GB of RAM, and 2.8 GHz CPU. Also, the 3 metaheuristic algorithms (WOA, PSO, and FA) are coded in MATLAB R2012a.

6.2. Analysis of the robust RAS model

In this section, we test the proposed robust RAS model through a set of numerical experiments implemented on the case study presented in 6.1. The purpose of the numerical experiments is three-fold. Our first goal is to analyze the trade-off between the level and the cost of robustness. More precisely, we are interested to see how increasing the budget of uncertainty affects the feasibility and optimality of the deterministic schedule. This is important for the decision maker in order to set a budget of uncertainty so that the plan obtained by the robust model is feasible in the presence of future uncertainties while it is not too costly (i.e., overprotected against uncertainty). This goal

is achieved through comparison between the objective function value of the robust model and the nominal one for different levels of budget of uncertainty. Our second goal is to verify the feasibility of the robust optimal schedule in the presence of randomly generated uncertain treatment times. Monte Carlo simulation is employed for this purpose because it is a more realistic approach for evaluating the impact of the budget of uncertainty on the feasibility and the cost of the schedule. We finally compare the objective function value of the robust RAS model with a worst-case scenario deterministic model to compare their degree of conservatism. This measure is used to verify whether or not the solution of robust model is overprotected against uncertainty.

The analyses mentioned above are carried out on four sets of test problems that are distinguished by the level of variability of uncertain parameters. More precisely, we define γ as the level of variability of uncertain treatment times comparing to their nominal values and consider four classes of test problems corresponding to γ = 5%,10%,20% and 30%. While considering uncertainty, we assume that Γ^1_{mkf} and Γ^2_{mkf} vary from 0 to $|WP|$ =40 (the worst-case) and Γ^3_{mkf} vary from 0 to $|BP|$ = 76. We also consider 0%, 10%, 30%, 50%, 70%, 90% and 100% of the maximum budget of uncertainty in our experimental results. Finally, the Monte-Carlo simulation is based on generating random scenarios for uncertain setup time and treatment times from their corresponding uniform distributions. Afterwards, for each scenario, the optimal solution of the robust model is plugged into the deterministic model where the uncertain parameters are substituted by the simulated value. Afterwards, the feasibility and the actual objective function value of the robust solution are verified.

6.2.1. Experimental results on the RAS model

As we mentioned in chapter 4, the uncertain treatment times affect the feasibility of constraint (37) in model (27) - (48) which is related to the machines capacity. Also, it is evident that such uncertainty has an impact on the number of newly scheduled patients from the waiting list; hence, it has an impact on the optimality of the deterministic schedule.

The trade-off between robustness (i.e., the budget of uncertainty) and the cost of robustness is estimated by calculating the objective value deviation $\frac{Z_R^N - Z_R^R}{Z_R^N}$, where Z_R^N and Z_R^R are the nominal and robust optimal objective values, respectively.

Figure 6- The trade-off between budget of uncertainty and objective function value in robust RAS model

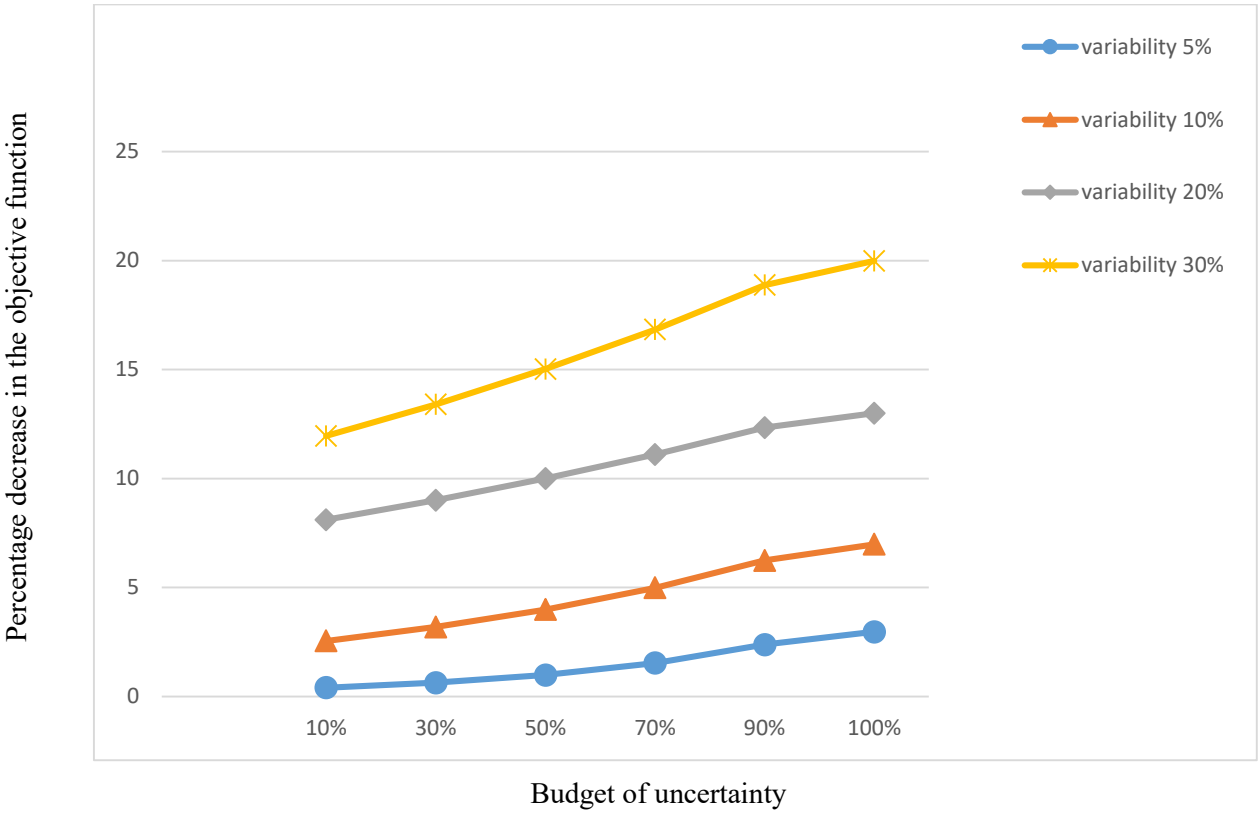
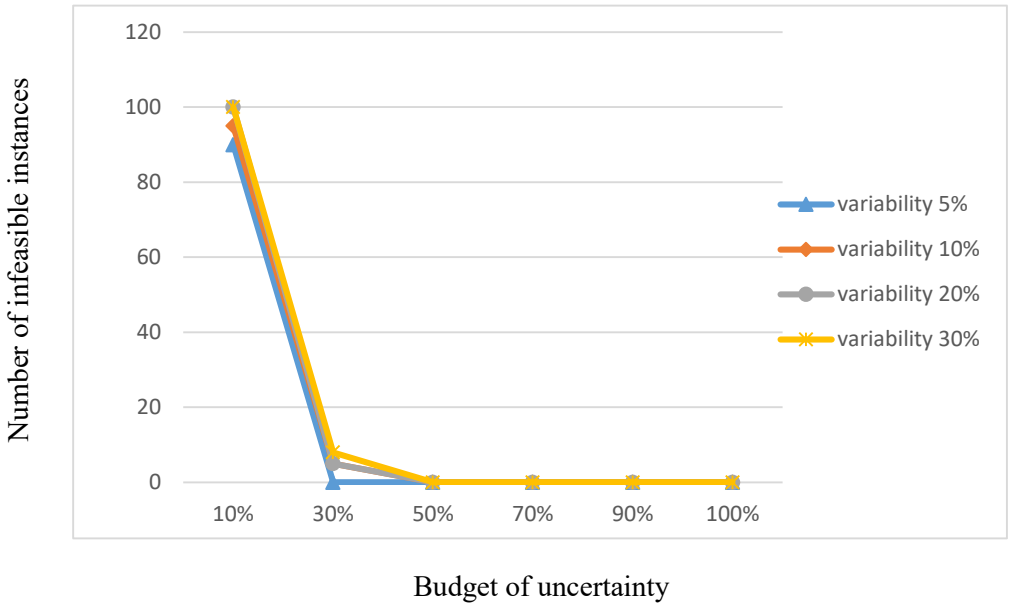


Figure 6 represents the percentage decrease in the objective function value versus the nominal one (i.e., objective function deviation) for four levels of treatment time variability and different values of the budget of uncertainty. As expected, when robustness is enforced to the model, the objective function (i.e., the number of newly scheduled patients) is decreased in order to envisage worst-case values for a certain number of treatment time parameters in the capacity constraint (constraint 37). The decrease in the robust objective function value is the effect of considering such worst-case time values on scheduling patients appointments. It is worth noting that such worst cases occur when time parameters take their highest value. In such cases, fewer patients are scheduled. Additionally, we can conclude from Figure 6, when the variability level of uncertain parameters is small (i.e., 5%), the impact of imposing robustness on the objective function is less significant in comparison with the higher level of variability.

On the other hand, when the budget of uncertainty is increased, it indicates that the number of uncertain parameters that take their worst-case value are increased which in turn, is expected to decrease the number of newly scheduled patients.

Another important issue is the analysis of the robust schedule by considering a given budget of uncertainty (less than the maximum budget) is associated with its feasibility. In other words, when the budget of uncertainty achieves its maximum value, the robust solutions are always feasible. In contrast, the feasibility condition cannot be guaranteed by considering smaller values for the budget of uncertainty. In order to resolve this issue, we conduct a set of Monte-Carlo simulation experiments, where we randomly generate 100 random treatment times based on different variability levels for each value of budget of uncertainty. Afterwards, we solve the deterministic RAS model for each scenario, where the schedule obtained from the robust RAS model is plugged into the aforementioned deterministic models. Consequently, we solved 100×6×4 = 2400 deterministic models corresponding to each randomly generated scenario. Through such experiments, we first verify the feasibility of the robust solution in the deterministic problem with simulated random treatment times. These results are depicted in Figure 7 for the four classes of test instances.

Figure 7- Feasibility of robust solutions with simulated treatment times



As is shown in Figure 7, when the budget of uncertainty is greater than 50%, the number of infeasible instances equal to zero. Thus, by considering $\Gamma \geq 50\%$, it is possible to guarantee the feasibility of solutions for random treatment times, and it is not necessary to increase the value of Γ and enforce less assignments. This will cause less than 15% decrease in objective value for all

of the variabilities. Hence, the robust solution with 50% budget of uncertainty has the high quality in terms of feasibility with an acceptable underestimation of the patients' assignment.

Finally, we compare the robust problem with the worst-case deterministic model (WC). Z^{WC} denotes the optimal nominal objective value when the deterministic nominal model is solved by considering the worst-case bound in the given uncertain interval. If $(Z_R^R - Z^{WC}) \geq 0$, it can be concluded that the robust problem proposes a solution with higher number of newly scheduled patients compared with the worst-case deterministic model. The latter analysis is presented in Table (55).

Table 55- Comparison of Z^{WC} and Z_R^R

	γ=5%	γ =10%	γ =20%	γ =30%
Z^{WC}	1,373,566,128 (18 patients)	1,373,059,835 (18 patients)	1,372,503,424 (16 patients)	1,372,503,233 (16 patients)
Z_R^R	1,569,350,701 (29 patients)	1,382,439,028 (22 patients)	1,378,131,711 (19 patients)	1,375,955,015 (19 patients)

Since $(Z_R^R - Z^{WC}) \geq 0$ for all variability levels in Table (55), it is evident that the robust RAS model outperforms the worst-case deterministic problem in terms of the weighted number of newly scheduled patients.

6.3. Application of metaheuristics to radiotherapy appointment scheduling model

In this section, we present the results of implementing the 3 meatheuristics, discussed in chapters 3 and 5, on deterministic and robust RAS models. The performance of the aforementioned algorithms is validated via comparison with the results of a commercial solver (CPLEX 12.6.1). Table 56 depicts population size and maximum number of iterations exploited in each algorithm. Afterwards, we discuss the results of implementing these algorithms on 2 test instances that differ in terms of the number of machines (2 and 3, respectively).

Table 56- Maximum population and iteration size of metaheuristics

	Population size (I_{max})	Maximum number of iterations (t_{max})
WOA	10	15
PSO	10	15
FA	10	15

Table (56) summarizes the objective value, CPU time, and optimality gap obtained after implementing each metaheuristic. It is worth noting that the optimality gap is calculated as the relative difference between the optimal objective function value obtained by CPLEX and the converged solution of metaheuristics. It is noteworthy that CPLEX can provide optimal solutions in the second test instance for both deterministic and robust RAS models in 4,212.19 and 11,427.60 seconds, respectively.

Table 57- Metaheuristics results on two test instances

				CPLEX	WOA	PSO	FA
Deterministic RAS model		instance 1	Objective function value	1,548,757,647	1,547,193,937	1,536,264,406	1,547,380,290
			optimality gap		0.001009	0.008066	0.000889
			CPU time(seconds)	76.85	64.8945	66.02	373.53
		instance 2	Objective function value	1,553,374,602	1,548,714,612	1,548,408,930	1,547,976,496
			optimality gap		0.002999	0.003196	0.003475
			CPU time(seconds)	4212.19	88.2515	92.0928	504.9971
Robust RAS model	Variability=10% gama=0.5	instance 1	Objective function value	1,523,622,370	1,471,899,608	1,449,556,953	1,519,245,428
			optimality gap		0.033947	0.048611	0.002872
			CPU time(seconds)	156.63	78.1048	75.8457	428.2536
		instance 2	Objective function value	1,553,326,619	1,481,060,707	1,517,052,600	1,519,274,301
			optimality gap		0.046523	0.023352	0.021922
			CPU time(seconds)	11427.60	112.8926	101.9051	578.4286

As it can be observed in table (57), all metaheuristics provide high quality feasible schedules (lower bounds) with an average optimality gap of 1.6% and 1.7% in test instances 1 and 2 for deterministic and robust RAS models, respectively. Also, the CPU time of WOA and PSO are also smaller than the solver (94.6131% in average). This CPU time gap is more significant for the robust RAS model that includes higher number of decision variables and constraints. The FA algorithm, on the contrary converges slowly despite the high quality of its solution. This can be explained by the large number of operations carried out in each iteration of this algorithm due to investigating all possible neighborhoods for improving current schedule. In other words, in each iteration of FA, the position of all pairs of fireflies are updated according to the best one. However,

in PSO this is done only according to the best particle and best global solution. Also, in WOA updates are done towards either the best position or a random one.

Figures 8-19 depict the improvement in the objective function at each iteration of PSO, WOA, and FA metaheuristics in deterministic and robust RAS models for instances 1 and 2. As it can be observed in figures 8-9, for both instances, both PSO and WOA algorithms converge to a high quality feasible solution in 5 iterations while applied to the deterministic RAS model. On the contrary, the robust RAS model includes more decision variables and constraints; hence in this case (figures 11-12), theses algorithms converge slightly slower.

Figure 8- PSO performance in deterministic RAS model (instance 1)

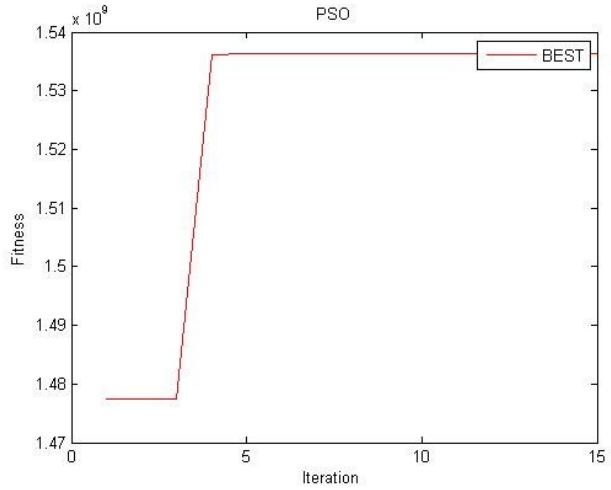


Figure 9- WOA performance in deterministic model (instance 1)

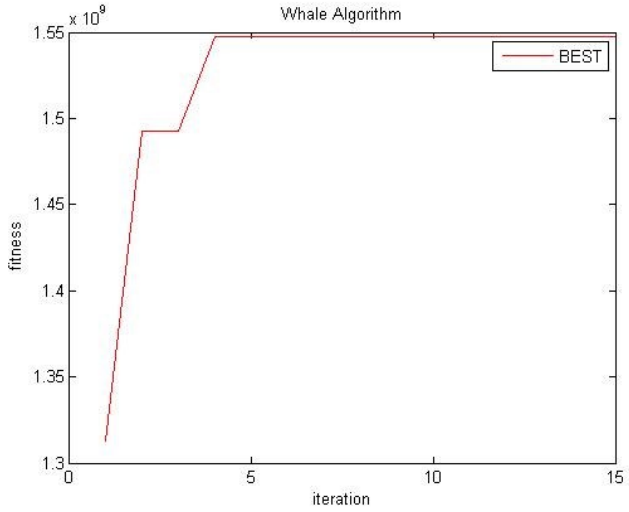


Figure 10- FA performance in deterministic model (instance 1)

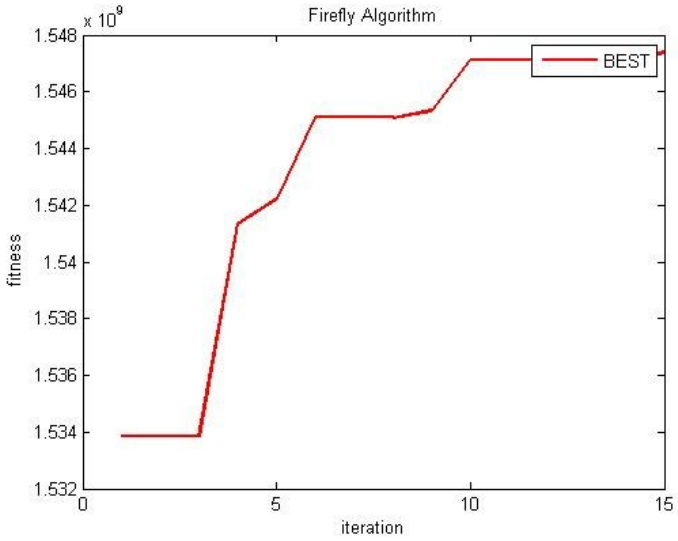


Figure 11- PSO performance in robust RAS model (instance 1)

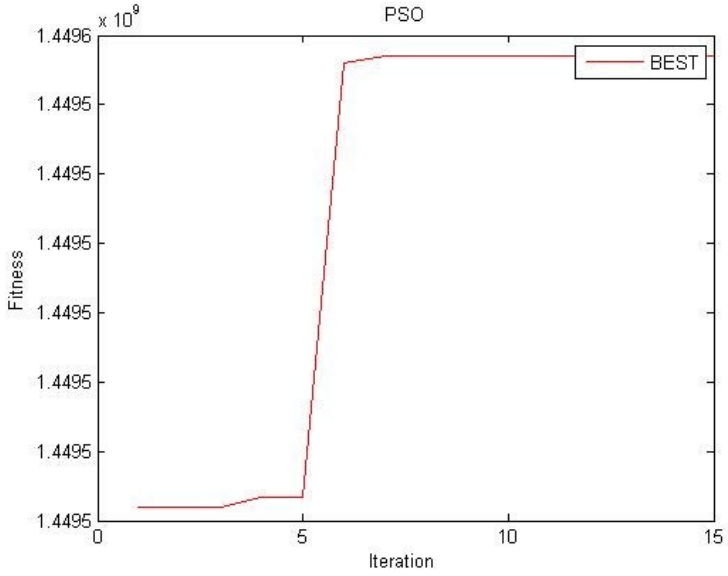


Figure 12- WOA performance in robust RAS model (instance 1)

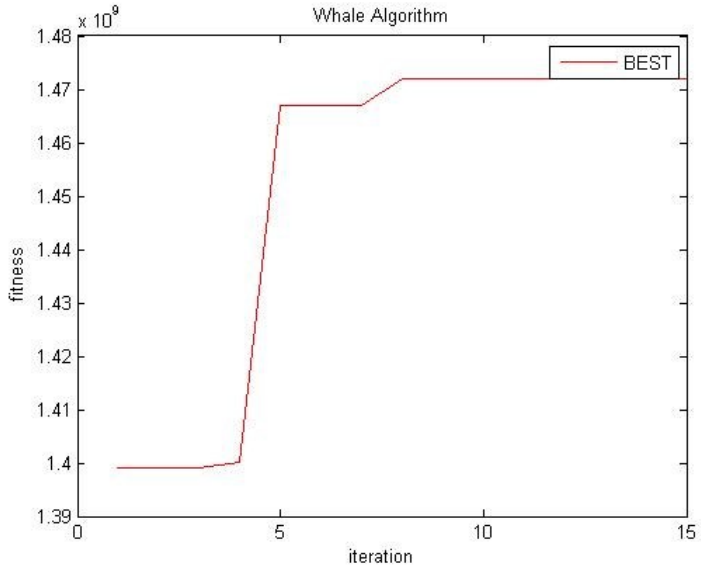


Figure 13- FA performance in robust RAS model (instance 1)

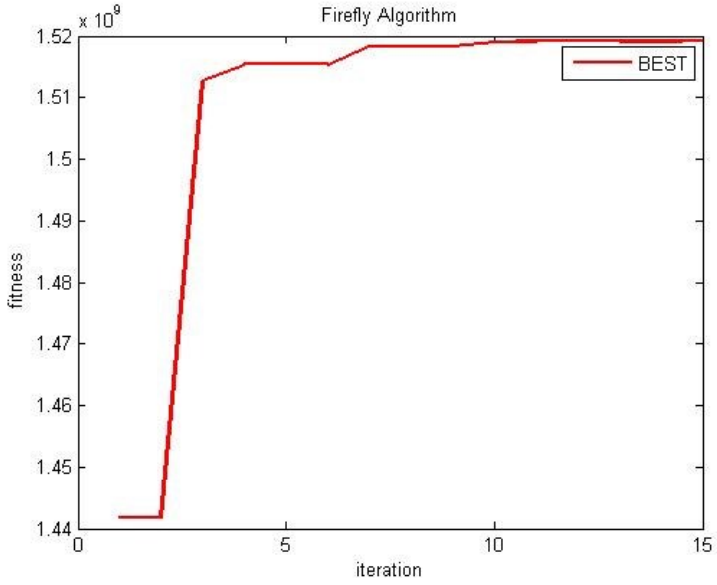


Figure 14-WOA performance in deterministic model (instance 2)

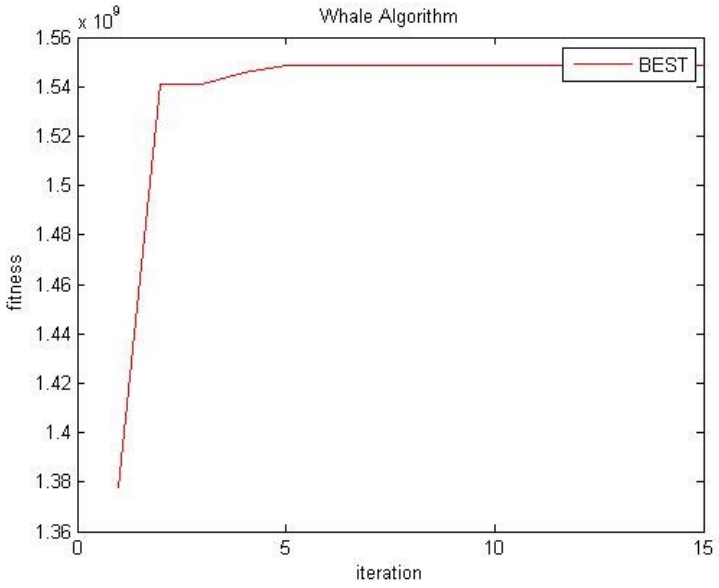


Figure 15- PSO performance in deterministic model (instance 2)

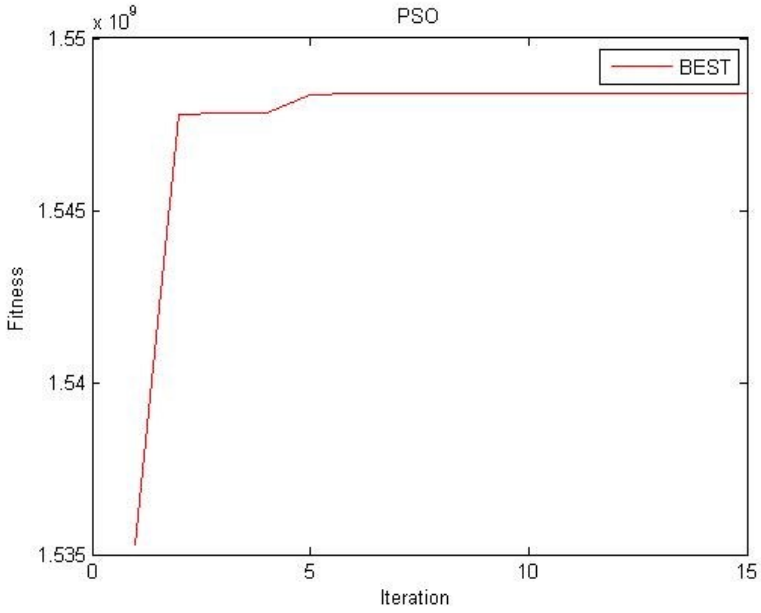


Figure 16- PSO performance in deterministic model (instance 2)

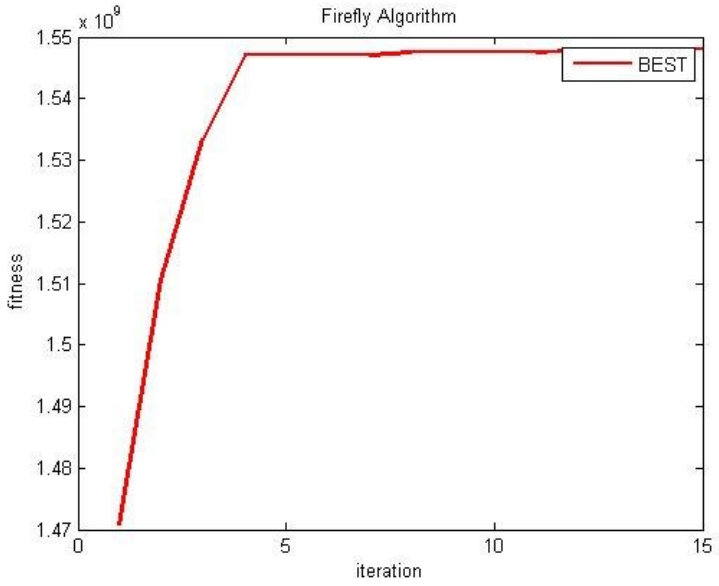


Figure 17- WOA performance in robust RAS model (instance 2)

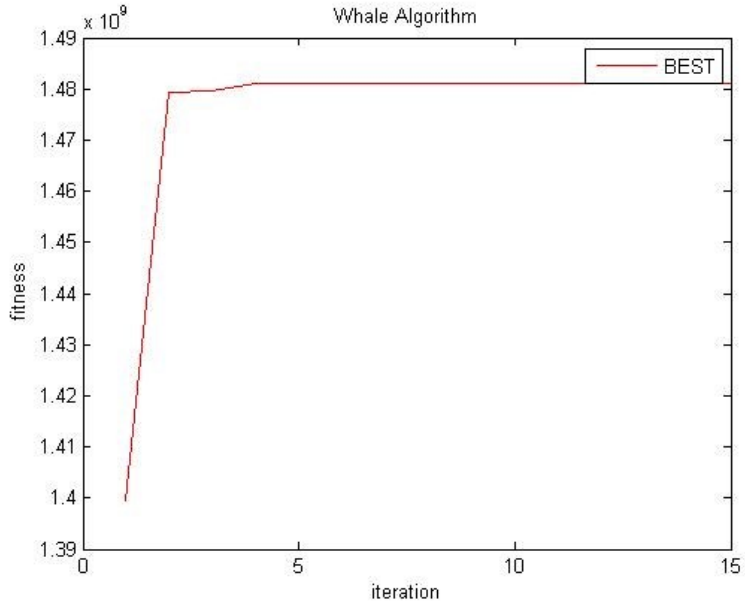


Figure 18- PSO performance in robust RAS model (instance 2)

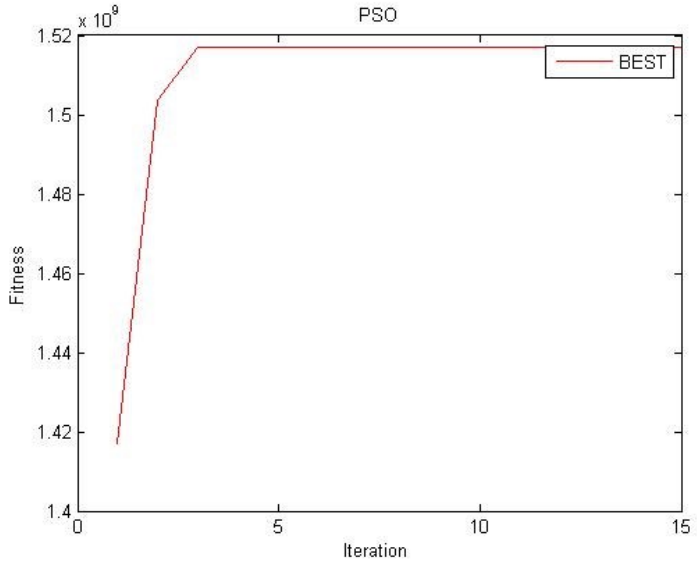
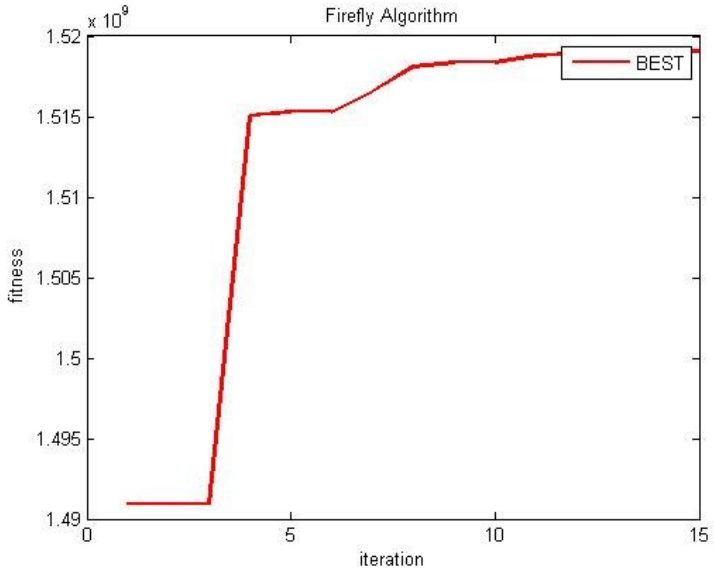


Figure 19- FA performance in robust RAS model (instance 2)



From the above experimental results, it can be concluded that CPLEX could obtain an optimal solution for the test instance with two machines in 76.85 and 156.63 seconds for deterministic and robust RAS models, respectively. Nonetheless, for larger number of machines (test instance 2) it can obtain an optimal solution in 4,212.19 and 11,427.6 seconds for deterministic and robust RAS models, respectively. In the first test instance, the WOA and PSO metaheuristics outperform CPLEX regarding the CPU time. Nonetheless, for the larger test instance (instance 2), the CPU

time of all metaheuristics are significantly smaller than CPLEX. Among the 3 metaheuristics, WOA seems to be the fastest algorithm followed by PSO and FA. Despite the slow convergence rate, FA outperforms the two other metaheuristics in terms of quality of lower bound (optimality gap) due to a full neighborhood search in each iteration. These results could be further validated through substantially larger problem instances.

Chapter 7: Conclusion and future research

Motivated by several challenges radiotherapy clinics are facing in order to schedule new patients from a waiting list while dealing with a large number of patients who have already started their treatments, in this thesis, we investigated an offline radiotherapy appointment scheduling problem. In order to maximize the number of scheduled patients, we assumed that the patients who have already started their treatment in a previous planning horizon (booked patients) would accept to be rescheduled during the current horizon. Furthermore, to add more flexibility, we also incorporated treatment time windows for both groups of booked and new patients. Such time windows represent the earliest, due, and latest dates where those patients could receive treatments according to their treatment plan. The problem was then formulated as a deterministic mixed-integer-program with the objective of maximizing the number of newly scheduled patients subject to the continuity of treatments on the same machine per patient, capacity of machines, treatment duration, and patients' priorities as some constraints among others. On the other hand, patients in these clinics require variable treatment times depending on the type, stage, and severity of cancer. Such variability might make the schedule determined by a deterministic radiotherapy appointment scheduling (RAS) model infeasible in terms of available capacity of radiotherapy machines during regular daily shifts. In other words, some patients won't be able to receive treatments due to unavailability of machines. Hence, with the goal of protecting the schedule against the treatment time perturbations, we reformulated the above-mentioned RAS model as a cardinality-constrained robust optimization model. This approach provides the possibility to define a budget of uncertainty and accordingly find a trade-off between schedule robustness (i.e., protection of feasibility of schedule in the presence of high treatment times) and the cost of robustness (i.e., the reduction in the number of newly scheduled patients).

Both deterministic and robust RAS models incorporate high number of binary decision variables that make them hard for a commercial solver to find an optimal solution in reasonable time for large-scale instances with large number of machines and long patient lists. In order to alleviate this computational complexity, we proposed three swarm-based, nature-inspired metaheuristics, namely whale optimization algorithm, particle swarm optimization, and firefly algorithm.

Finally, we designed a case study inspired from a real radiotherapy clinic with the goal of validating the proposed robust RAS model and the three metaheuristics. The first part of our numerical experiments, conducted as a set of Monte-Carlo simulation tests, revealed that by considering half of treatment times at their worst-case scenario, the feasibility of schedule can be guaranteed in the presence of a large number of randomly generated scenarios without a significant reduction in the number of newly scheduled patients. Although this result is valid for the case study under investigation, the proposed RAS model and Monte-Carlo simulation platform provide the possibility to the decision-maker to evaluate the feasibility and cost of schedule for different budgets of uncertainty. Finally, our numerical results on the application of the three proposed metaheuristics revealed that these algorithms provide high quality schedules with negligible optimality gap at a significantly reduced CPU time, for two realistic instances, comparing with a commercial solver.

As a general remark, the proposed robust RAS model and the solution algorithms are expected to enable radiotherapy clinics to provide more reliable and efficient treatment services. Further, as possible exploitation, the robust model and algorithms can be properly implemented and embedded into a software system that could be one of the core modules of a more sophisticated decision support system able to organize and manage all the clinical procedures of a radiotherapy clinic in a hospital. Also, current model and solution approaches can be further validated through more case instances that incorporate more machines and patients. Future research could also focus on the integration of proposed offline scheduling tool with an online scheduling module with the goal of maximizing the utilization of machines/staff in the presence of patient no-shows and maximising the number of scheduled patients who are referred to the clinic as emergency cases.

References

Ahmadi-Javid, A., Jalali, Z. & Klassen, K.J., 2016. Outpatient Appointment Systems in Healthcare: A Review of Optimization Studies. *European Journal of Operational Research*.

Ben-Tal, A. & Nemirovski, A., 1998. Robust Convex Optimization. *Mathematics of Operations Research*, 23(4), pp.769–805.

Bertsimas, D. & Sim, M., 2004. The Price of robustness. *Operations Research*, 52(1), pp.35–53. Available at: http://www.robustopt.com/references/Price of Robustness.pdf.

Burke, E.K., Leite-Rocha, P. & Petrovic, S., 2011. An Integer Linear Programming Model for the Radiotherapy Treatment Scheduling Problem. , (2007), pp.1–24.

Castro, E. & Petrovic, S., 2012. Combined mathematical programming and heuristics for a radiotherapy pre-treatment scheduling problem. *Journal of Scheduling*, 15(3), pp.333–346.

Cayirli, T. & Veral, E., 2003. Outpatient scheduling in health care: A review of literature. *Production and Operations Management*, 12(4), pp.519–549.

Conforti, D. et al., 2009. An optimal decision-making approach for the management of radiotherapy patients. *OR Spectrum*, 33(1), pp.123–148.

Conforti, D., Guerriero, F. & Guido, R., 2010. Non-block scheduling with priority for radiotherapy treatments. *European Journal of Operational Research*, 201(1), pp.289–296.

Conforti, D., Guerriero, F. & Guido, R., 2008. Optimization models for radiotherapy patient scheduling. *4or*, 6(3), pp.263–278.

Eberhart, R. & Kennedy, J., 1995. A New Optimizer Using Particle Swarm Theory.

Gupta, Diwakar;Denton, B., 2008. Appointment scheduling in health care: Challenges and opportunities. *IIE Transactions*, 40(9), pp.800–819.

Kapamara, T. et al., 2006. A review of scheduling problems in radiotherapy. *Proceedings of the International Control Systems Engineering Conference (ICSE)*, (September), pp.207–211.

Kapamara, T. & Petrovic, D., 2009. A heuristics and steepest hill climbing method to scheduling radiotherapy patients. *Proceedings of the International Conference on Operational*

Research Applied to Health Services (ORAHS), (December).

Kapamara, T., Sheibani, K. & Petrovic, D., 2007. A simulation of a radiotherapy treatment system: A case study of a local cancer centre. *Proc. of ORP3 ...*, pp.1–7.

Kolisch, R. & Sickinger, S., 2008. Providing radiology health care services to stochastic demand of different customer classes. *OR Spectrum*, 30(2), pp.375–395.

Liang, B. et al., 2014. Improvement of chemotherapy patient flow and scheduling in an outpatient oncology clinic. *International Journal of Production Research*, 7543(July 2015), pp.1–14.

Marie-andr, A.L. & Rousseau, N.L.L., 2015. Online stochastic optimization of radiotherapy patient scheduling. , pp.110–123.

Mirjalili, S. & Lewis, A., 2016. The Whale Optimization Algorithm. *Advances in Engineering Software*, 95, pp.51–67.

Ogulata, S.N. et al., 2009. A simulation approach for scheduling patients in the department of radiation oncology. *Journal of Medical Systems*, 33(3), pp.233–239.

Osman, I.H. & Kelly, J.P., 1996. Meta-Heuristics: An Overview. In *Meta-Heuristics*. Boston, MA: Springer US, pp. 1–21.

Palupi Rini, D., Mariyam Shamsuddin, S. & Sophiyati Yuhaniz, S., 2011. Particle Swarm Optimization: Technique, System and Challenges. *International Journal of Computer Applications*, 14(1), pp.19–27.

Pérez, E. et al., 2016. IIE Transactions on Healthcare Systems Engineering Patient and resource scheduling of multi-step medical procedures in nuclear medicine Patient and resource scheduling of multi-step medical procedures in nuclear medicine. , 8300(July).

Pérez, E. et al., 2013. Stochastic online appointment scheduling of multi-step sequential procedures in nuclear medicine. *Health Care Management Science*, 16(4), pp.281–299.

Petrovic, D., Morshed, M. & Petrovic, S., 2009. Genetic Algorithm Based Scheduling of Radiotherapy Treatments for Cancer Patients. *Proceedings of the Conference on Artificial Intelligence in Medicine (AIME)*, 5651, pp.101–105.

Petrovic, D., Morshed, M. & Petrovic, S., 2011. Multi-objective genetic algorithms for scheduling of radiotherapy treatments for categorised cancer patients. *Expert Systems with Applications*, 38(6), pp.6994–7002.

Petrovic, S. & Leite-Rocha, P., 2008. Constructive Approaches to Radiotherapy Scheduling. *World Congress on Engineering and Computer Science (WCECS)*, pp.722–727.

Proctor, S. et al., 2007. Modelling Patient Flow in a Radiotherapy Department. *OR Insight*, 20(3), pp.6–14.

Sauré, A. et al., 2012. Dynamic multi-appointment patient scheduling for radiation therapy. *European Journal of Operational Research*, 223(2), pp.573–584.

Yang, X.-S., 2010a. *Engineering Optimization An Introduction with Metheuristic Application*,

Yang, X.-S., 2010b. Firefly Algorithm, Lévy Flights and Global Optimization. In *Research and Development in Intelligent Systems XXVI*. London: Springer London, pp. 209–218.

Appendix

9.1. Fitness function MATLAB code

```
function sol=fitness(sol,data)
load data
pos=sol.x;

%assigns random numbers in [0, 1] to solution matrix elements

pos(8,:)=CB(pos(8,:),0,1);

%assigns infinity value to elements beyond indices
pos(1,np+1:end)=inf;
pos(2,nk+1:end)=inf;
pos(3,nf+1:end)=inf;
pos(4,nm+1:end)=inf;
pos(5,nj+1:end)=inf;
pos(6,nk+1:end)=inf;
pos(7,nf+1:end)=inf;

%decision variable definition
ZP=zeros(np,nk,nf);
YP=zeros(np,nk,nf);
SSS=zeros(nm,nk,nf);
SSS0=zeros(nm,nk,nf);
CH=zeros(1,18);

%sorts rows to determine a permutation for p, k, and f
[~,P]=sort(pos(1,:));P=P(1:np);
[~,K]=sort(pos(2,:));K=K(1:nk);K=nk:-1:1;
[~,F]=sort(pos(3,:));F=F(1:nf);

%these three loops check constraints related to z_pkf and y_pkf variables to assign them 0 and 1 values
for p=P
   for k=K
      for f=F

         syp1(:,:)=sum(YP,3);
         C14L=sum(sum(YP.*av,3),2);

         m=1:nm;
         A=YP(p,k,f).*av(p,k,f);

         C10L=sum(Sp(:,p).*A,2);C10L=C10L<T(:,k,f);

         SSS(m,k,f)=sum(Sp(:,p).*A,2);

         if k<=dp(p) && av(p,k,f)==1 && syp1(p,k)==0 && C14L(p)<tp(p) && sum(C10L)==nm
            YP(p,k,f)=1;
         end
```

```
        ZPP=ZP;ZPP(p,k,f)=1;

        zp1=ZP(p,1:idp(p),:);
        C12L=sum(sum(zp1,3),2);



        C15=1;
        if k<=idp(p)
           dk=k+1:k+tp(p)-1;
           A=[];A=YP(p,dk,:).*av(p,dk,:);C15l=sum(A(:));
           A=(tp(p)-1).*ZPP(p,k,:);C15r=sum(A(:));
           if C15l<C15r;C15=0;end
        end

        A=ZPP(p,1:idp(p),:).*av(p,1:idp(p),:);C12=sum(A(:));
        pkf=[p k f];
        rp(p):nk;
        1:idp(p);
        aa=[];aa(:,:)=av(p,1:idp(p),:);

        if k>=rp(p) && YP(p,k,f)==1 && C12L==0 && C15==1
           ZP(p,k,f)=1;
        end







     end


     if k<=idp(p)
        dk=k+1:k+tp(p)-1;
        A=[];A=YP(p,dk,:).*av(p,dk,:);C15l=sum(A(:));
        A=(tp(p)-1).*ZP(p,k,:);C15r=sum(A(:));
        CH(15)=CH(15)+max(C15r-C15l,0);
     end

  end
end



tmj=repmat(tj',nm,1);
SS1=repmat(S1,[1,1,nk,nf]);
SSj=repmat(Sj,[1,1,nk,nf]);

%decision variables definition
Z=zeros(nm,nj,nk,nf);
Y=zeros(nm,nj,nk,nf);
```

```
%define permutation for last row
x8=pos(8,:);x8=x8*2;x8=x8-1;x8(x8<0)=0;x8=ceil(x8.*nm);

%decision variables definition

X=zeros(nm,nj);
SZT=zeros(nm,nj);


%define permutation for m, j, k, f

[~,M]=sort(pos(4,:));M=M(1:nm);
[~,J]=sort(pos(5,:));J=J(1:nj);
[~,K]=sort(pos(6,:));K=K(1:nk);K=nk:-1:1;
[~,F]=sort(pos(7,:));F=F(1:nf);

%these four loops check the constraints related to x_mj, z_mjkf and y_mjkf to assign them 0 and 1 values
for m=M
   for j=J
      for k=K
         for f=F


            ZZ=Z;ZZ(m,j,k,f)=1;



            C9=1;
            C9L=sum(sum(Y(m,j,:,:),3),4)+sum(sum(ZZ(m,j,1:dj(j),:),3),4);
            C9R=tmj(m,j).*X(m,j);

            C10sy=sum(Y(m,:,k,f).*Sj(m,:));
            C10sz=sum(ZZ(m,:,k,f).*S1(m,:));
            SSS0(m,k,f)=C10sy+C10sz;

            C8=1;
            if k<=id(j)
               dk=k+1:k+tj(j)-1;
               A=Y(m,j,dk,:);C8l=sum(A(:));
               A=(tj(j)-1).*ZZ(m,j,k,:);C8r=sum(A(:));
               if C8l<C8r;C8=0;end
            end


            if m==x8(j);
               X(m,j)=1;
               SZ=sum(sum(Z(m,j,1:dj(j),:),3),4);
               if SZ==0 && X(m,j)==1 && k<=id(j) && k>=rj(j)
                  if  C8==1 && C9L<=C9R  && C10sy+C10sz+SSS(m,k,f)<=T(m,k,f)
                     Z(m,j,k,f)=1;
                  end
               end
            end
```

```
            YY=Y;YY(m,j,k,f)=1;


            C7L=[];C7L(:,:,:)=sum(YY+Z,4);
            C7=(C7L>repmat(X,[1,1,nk]));C7=any(C7(:));

            C8=1;
            if k<id(j)
               dk=k+1:k+tj(j)-1;
               A=YY(m,j,dk,:);C8l=sum(A(:));
               A=(tj(j)-1).*Z(m,j,k,:);C8r=sum(A(:));
               if C8l<C8r;C8=0;end
            end

            C9L=sum(sum(YY(m,j,:,:),3),4)+sum(sum(Z(m,j,1:dj(j),:),3),4);
            C9R=tmj(m,j).*X(m,j);

            C10sy=sum(YY(m,:,k,f).*Sj(m,:));


            C4L=sum(YY(m,j,k,:));

            if X(m,j)==1 && C8==1 && k<=dj(j) && C4L<=X(m,j) && C9L<=C9R &&
C10sy+SSS(m,k,f)<=T(m,k,f) && C7==0
               Y(m,j,k,f)=1;
            end




            C10sy=sum(Y(m,:,k,f).*Sj(m,:));
            C10sz=sum(ZZ(m,:,k,f).*S1(m,:));
            SSS0(m,k,f)=C10sy+C10sz;


            if k<=id(j)
               dk=k+1:k+tj(j)-1;
               A=Y(m,j,dk,:);C8l=sum(A(:));
               A=(tj(j)-1).*Z(m,j,k,:);C8r=sum(A(:));
               CH(8)=CH(8)+max(C8r-C8l,0);
            end




         end

         A=[];A=Y(m,j,k,:);CH(4)=CH(4)+sum(max(A-X(m,j),0));
```

```
        end

        SZT(m,j)=sum(sum(Z(m,j,1:dj(j),:),3),4);

    end
end

%calculates violation for each constraint
CH(1)=sum(sum(abs(SZT-X)));
CH(2)=sum(max(sum(X,1)-1,0));

for j=1:nj
    A=[];A=Z(:,j,id(j)+1:end,:);CH(3)=CH(3)+sum(A(:));
    A=[];A=Z(:,j,1:rj(j)-1,:);CH(5)=CH(5)+sum(A(:));
    A=[];A=Y(:,j,dj(j)+1:end,:);CH(6)=CH(6)+sum(A(:));
end

A=Z+Y;C7L=[];C7L(:,:,:)=sum(A,4);A=max(C7L-repmat(X,[1,1,nk]),0);CH(7)=sum(A(:));
A=sum(sum(Y,3),4)+sum(sum(Z,3),4);B=tmj.*X;CH(9)=sum(sum(abs(A-B)));
A=max(SSS0+SSS-T,0);CH(10)=sum(A(:));
A=max(YP-av,0);CH(11)=sum(A(:));
A=max(ZP-YP,0);CH(13)=sum(A(:));
A=max(sum(YP.*av,3)-1,0);CH(16)=sum(A(:));


for p=1:np
    A=[];A(:,:)=ZP(p,1:idp(p),:).*av(p,1:idp(p),:);CH(12)=CH(12)+0*abs(sum(A(:))-1);
    A=[];A=YP(p,:,:).*av(p,:,:);CH(14)=CH(14)+abs(sum(A(:))-tp(p));
    tpp(p)=sum(A(:));
    A=[];A=YP(p,dp(p)+1:end,:);CH(17)=CH(17)+sum(A(:));
    A=[];A=ZP(p,1:rp(p)-1,:);CH(18)=CH(18)+sum(A(:));

end



%objective function

obj=repmat(W',nm,1).*X;

SCH=sum(CH);



OBJ=sum(obj(:));
fit=1/OBJ;
%considering violations to correct and solutions
sol.fit=fit*(1+1000*SCH);
sol.info.X=X;
sol.info.Y=Y;
sol.info.YP=YP;
sol.info.Z=Z;
sol.info.ZP=ZP;
```

```
sol.info.X=X;
sol.info.OBJ=OBJ;
sol.info.CH=CH;
sol.info.CH=CH;
sol.info.SCH=SCH;
sol.info.C10L=SSS0+SSS;
sol.info.SZT=SZT;
sol.info.C10=SSS0+SSS;
end
```

9.2. WOA MATLAB code

```
clc
clear
close all
format shortG

%% Parameters Definiterion
data=InsertData();load data

lb=0*ones(8,nvar);
ub=1*ones(8,nvar);

npop=10;
maxiter=15;


data.lb=lb;
data.ub=ub;
%% Initialization
tic
emp.x=[];
emp.info=[];
emp.fit=[];
emp.info.SCH=[];

pop=repmat(emp,npop,1);



for i=1:npop
pop(i).x=unifrnd(lb,ub);
pop(i)=fitness(pop(i),data);
end


[~,ind]=min([pop.fit]);
gpop=pop(ind);

Size.x=size(pop(1).x);


%% Main loop
BEST=zeros(maxiter,1);
MEAN=zeros(maxiter,1);


for iter=1:maxiter


   for i=1:npop
```

```
    % Return back the search agents that go beyond the boundaries of the search space
    Flag4ub=pop(i).x>ub;
    Flag4lb=pop(i).x<lb;
    pop(i).x=(pop(i).x.*(~(Flag4ub+Flag4lb)))+ub.*Flag4ub+lb.*Flag4lb;

    % Calculate objective function for each search agent
     pop(i)=fitness(pop(i),data);

    % Update the leader
    if pop(i).fit<gpop.fit
       gpop=pop(i); % Update alpha
    end

end

a=2-iter*((2)/maxiter); % a decreases linearly from 2 to 0

% a2 linearly decreases from -1 to -2 to calculate t
a2=-1+iter*((-1)/maxiter);

% Update the Position of search agents
for i=1:npop
    r1=rand(); % r1 is a random number in [0,1]
    r2=rand(); % r2 is a random number in [0,1]

    A=2*a*r1-a;   %equation 10
    C=2*r2;      %equation 11



    b=1;             %parameter b
    l=(a2-1)*rand+1;   %parameter l

    p = rand();       %probability

    for j=1:Size.x(2)

        if p<0.5
            if abs(A)>=1                    %exploration phase
                k =randi([1 npop]);
                D_X_rand=abs(C*pop(k).x(:,j)-pop(i).x(:,j)); %equation 8
                pop(i).x(:,j)=pop(k).x(:,j)-A*D_X_rand;      %equation 9

            elseif abs(A)<1                 %exploitation shrinking mechanism phase
                D_Leader=abs(C*gpop.x(:,j)-pop(i).x(:,j)); %equation 12
                pop(i).x(:,j)=gpop.x(:,j)-A*D_Leader;      %equation 13
            end

        elseif p>=0.5          %exploitation spiral mechanism

            distance2Leader=abs(gpop.x(:,j)-pop(i).x(:,j));

            pop(i).x(:,j)=distance2Leader*exp(b.*l).*cos(l.*2*pi)+gpop.x(:,j)%equation 14
```

```
        end

      end
    end


    BEST(iter)=gpop.info.OBJ;
    MEAN(iter)=mean([pop.fit]);

    FL=' Feasible'; if gpop.info.SCH>0;FL=' Infeasible';end

    disp(['iter ' num2str(iter) ' Best= ' num2str(BEST(iter)) FL]);



end

%% Results

disp(' ')
disp([ ' BEST fitness = '  num2str(BEST(iter))]);
disp([ ' Time = '  num2str(toc)]);

figure()
plot(BEST(1:iter),'r')

xlabel(' iteration ')
ylabel(' fitness')
legend( 'BEST')
title('Whale Algorithm')


write4D(gpop.info.Z,'Z')
write4D(gpop.info.Y,'Y')
write3D(gpop.info.ZP,'ZP')
write3D(gpop.info.YP,'YP')
write3D(gpop.info.C10,'C10')
write2D(gpop.info.X,'X')
write2D(gpop.info.SZT,'SumZ')

info=gpop.info;save out
Z=info.Z;
Y=info.Y;
ZP=info.ZP;
YP=info.YP;
X=info.X;
SZT=info.SZT;
```

9.3. PSO MATLAB code

```
clc
clear
close all
format shortG
%% Parameters Setting

data=InsertData();load data

lb.x=0*ones(8,nvar);
ub.x=1*ones(8,nvar);


lb.v=-0.8;
ub.v=0.8;


Npar=10; % Population Size
Maxiter=15; % Max Iteration

W=1;
C1=2;
C2=2;

W_RF=0.97;

FinalBEST=-1;


%% Initial Population
tic
emp.x=[];
emp.v=[];
emp.fit=[];
emp.info=[];
par=repmat(emp,Npar,1);


for i=1:Npar
 par(i).x=unifrnd(lb.x,ub.x);
 par(i).v=0;
 par(i)=fitness(par(i),data);
end


bpar=par;

[~,ind]=min([par.fit]);
gpar=par(ind);
```

```
%% Main Loop Pso

BEST=zeros(Maxiter,1);
MEAN=zeros(Maxiter,1);


for iter=1:Maxiter


  for i=1:Npar

 % Update Velocity
        par(i).v=W*par(i).v+...
        C1*rand(size(lb.x)).*(bpar(i).x-par(i).x)+...
        C2*rand(size(lb.x)).*(gpar.x-par(i).x);

        par(i).v=CB(par(i).v,lb.v,ub.v);

 % Update Position

        par(i).x=par(i).x+par(i).v;
        par(i).x=CB(par(i).x,lb.x,ub.x);

 % Call Fitness
        par(i)=fitness(par(i),data);


 % Update gpar and bpar

 if par(i).fit<bpar(i).fit
    bpar(i)=par(i);
 end

 if par(i).fit<gpar.fit
    gpar=par(i);
 end

  end



BEST(iter)=gpar.info.OBJ;
MEAN(iter)=mean([bpar.fit]);

   FL=' Feasible'; if gpar.info.SCH>0;FL=' Infeasible';end


disp([ 'Iter = ' num2str(iter) ' BEST = ' num2str(BEST(iter)) FL])
```

```
W=W*W_RF;
 end
%% Results

BEST=BEST(1:iter);
MEAN=MEAN(1:iter);


disp([ ' Best Fitness = ' num2str(gpar.fit) ])
disp([ ' Time = ' num2str(toc) ])


figure(1)
plot(BEST,'r')


xlabel('Iteration ')
ylabel(' Fitness ')
legend('BEST')
title('PSO')
gpop=gpar;
write4D(gpop.info.Z,'Z')
write4D(gpop.info.Y,'Y')
write3D(gpop.info.ZP,'ZP')
write3D(gpop.info.YP,'YP')
write3D(gpop.info.C10,'C10')
write2D(gpop.info.X,'X')
write2D(gpop.info.SZT,'SumZ')

info=gpop.info;save out
Z=info.Z;
Y=info.Y;
ZP=info.ZP;
YP=info.YP;
X=info.X;
SZT=info.SZT;
```

9.4. FA MATLAB code

```
clc
clear
close all
format shortG

%% Parameters Definiterion
data=InsertData();load data

lb=0*ones(8,nvar);
ub=1*ones(8,nvar);

maxiter=15;         % Maximum Number of iterations

npop=10;            % Number of Fireflies


L=1;
gamma=1./sqrt(L);          % Light Absorption Coefficient

beta0=0.5;                 % Attraction Coefficient Base Value

alpha=0.9;                 % Mutation Coefficient

alpha_RF=0.95;             %Radius Reduction Factor


%% Initialization
tic
emp.x=[];
emp.SCH=[];
emp.info=[];
emp.fit=[];

pop=repmat(emp,npop,1);



for i=1:npop
pop(i).x=unifrnd(lb,ub);
pop(i)=fitness(pop(i),data);
end


[~,ind]=min([pop.fit]);
gpop=pop(ind);
```

```
%% Main Loop

BEST=zeros(maxiter,1);
MEAN=zeros(maxiter,1);

for iter=1:maxiter

   newpop=pop;

   k=npop;

   for i=1:npop
      for j=1:npop
         if pop(j).fit<=pop(i).fit
            k=k+1;
            newpop(i)=pop(i);
            newpop(i).x=MoveSol(pop(i).x,pop(j).x,alpha,beta0,gamma,lb,ub);
            newpop(i)=fitness(newpop(i),data);
            newpop(k)=newpop(i);

         end
      end
   end

   % Merge
   [pop]=[pop;newpop;gpop];




   % Sort and Select
   [~, ind]=sort([pop.fit]);
   pop=pop(ind);
   pop=pop(1:npop);

   % Select Best Sol
   gpop=pop(1);

   BEST(iter)=gpop.info.OBJ;
   MEAN(iter)=mean([pop.fit]);

    NO=' Feasible';

    if any([gpop.info.SCH]>0)
       NO=' Infeasible';
    end

   disp(['iter ' num2str(iter) ' Best= ' num2str(BEST(iter)) NO]);

   % Reduction Mutation Coefficient
   alpha=alpha*alpha_RF;
```

```
end

%% Results

disp(' ')
disp([ ' BEST fitness = '  num2str(gpop.fit)]);
disp([ ' Time = '  num2str(toc)]);

figure()
plot(BEST(1:iter),'r','LineWidth',2)

xlabel(' iteration ')
ylabel(' fitness')
legend( 'BEST')
title('Firefly Algorithm')

write4D(gpop.info.Z,'Z')
write4D(gpop.info.Y,'Y')
write3D(gpop.info.ZP,'ZP')
write3D(gpop.info.YP,'YP')
write3D(gpop.info.C10,'C10')
write2D(gpop.info.X,'X')
write2D(gpop.info.SZT,'SumZ')

info=gpop.info;save out
Z=info.Z;
Y=info.Y;
ZP=info.ZP;
YP=info.YP;
X=info.X;
SZT=info.SZT;
```