QATAR UNIVERSITY

COLLEGE OF ENGINEERING

OPTIMIZATION MODELS FOR MULTIPLE RESOURCE PLANNING

BY

NORAH MOHAMMED Z AL-DOSSARI

A Thesis Submitted to

the Faculty of the College of Engineering

in Partial Fulfillment of the Requirements for the Degree of

Masters of Science in Engineering Management

June 2021

COMMITTEE PAGE

The members of the Committee approve the Thesis of
Norah defended on 19/04/2021.

ABSTRACT

AL-DOSSARI, NORAH, MOHAMMED., Masters : June : 2021, Masters of Science in Engineering Management

Title: Optimization Models for Multiple Resource Planning

Supervisor of Thesis: Mohamed, Haouari, Mohamed Kharbeche.

Multiple resource planning is a very crucial undertaking for most organizations. Apart from reducing operational complexity, multiple resource planning facilitates efficient allocation of resources which reduces costs by minimizing the cost of tardiness and the cost for additional capacity. The current research investigates multiple resource loading problems (MRLP). MRLPs are very prevalent in today's organizational environments and are particularly critical for organizations that handle concurrent, time-intensive, and multiple-resource projects. Using data obtained from the Ministry of Administrative Development, Labor and Social Affairs (ADLSA), an MRLP is proposed. The problem utilizes data regarding staff, time, equipment, and finance to ensure efficient resource allocation among competing projects. In particular, the thesis proposes a novel model and solution approach for the MRLP. Computational experiments are then performed on the model. The results show that the model performs well, even in higher instances. The positive results attest to the effectiveness of the proposed MRLP problem.

DEDICATION

I would like to wholeheartedly dedicate this thesis to my beloved parents, husband, brothers, sisters, and my lovely daughters, who have been a source of strength and encouragement. I would also like to dedicate the thesis to my professors who offered guidance and support throughout the completion of the thesis.

Thanks for believing in me.

ACKNOWLEDGMENTS

I wish to express my sincere gratitude to the faculty and the thesis committee for their unwavering support and encouragement. Thank you for providing guidance during the completion of this research. The vision, experience, and sincerity motivated me to put more effort into the thesis. The learning opportunities offered to me were invaluable and will continue to shape my career going forward.

Completion of this thesis would not have been possible without the help and support of my professor. It has been an honor and a privilege to work hand-in-hand with such a sharp mind. I would specifically like to thank the professor for his understanding, empathy, and friendship. I would also like to extend my gratitude to my family and say thanks for their prayers and support. Lastly, to my caring and supportive husband: my sincere gratitude. Your words of encouragement throughout the research have been noted and deeply appreciated.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

1.1. Introduction

Efficient resource allocation and cost reduction are the ultimate goals for most high-level managers of companies. Over the last few decades, optimization models have emerged as effective tools for resource maximization and cost reduction for companies. Resource planning and optimization are particularly crucial for companies that handle multiple projects (Dooley et al. 2005). Project managers and planners in such companies face significant problems regarding the effective allocation of finances, staff, equipment, and other key resources to multiple projects. Optimization and planning models can be used in virtually every aspect of resource planning. The main objective is usually to ensure optimal resource allocation (revenue maximization, cost minimization, and resource sequence utilization) while adhering to constraints associated with resource availability. A key benefit of resource planning for companies is that it facilitates efficient decision-making. Resources in organizations can take many forms. These may be machines, equipment, crews, or vehicles. Resources facilitate the completion of tasks that may have specific due dates. Companies are required to meet specific due dates. Failure to complete tasks in time may result in losses (Rieck and J. Zimmermann, 2015). A lot of existing optimization models for resource planning provide a solution for single project optimization. However, with the increase in project complexities and the operation of multiple projects by companies, there is a need for the development and design of the optimization models for multiple resource planning (Rieck and J. Zimmermann, 2015). Although project problems are generalizable, companies that undertake single projects are likely to face different problems than those implementing multiple projects. Because of the importance of resource planning and optimization, optimization models have become subjects of significant interest.

1.2. Managing Multiple Projects

Multiple projects and resource planning are huge problems facing today’s organizations. Project managers no longer have to deal with a single project at a time. Like single project management, multiple project management also seeks to ensure that projects meet the stipulated requirements of performance, time, and budget. The management of multiple projects consists of key activities that include the formulation of project goals, design of the project planning and implementation, and project controlling (Rieck and J. Zimmermann, 2015). Most experts agree that the main issues associated with project management are lack of clear project goals, mismatch between project scope and project goals, poor identification and management of existing project issues, poor teamwork and coordination, and inefficient resource utilization.

Project problems become more pronounced when dealing with multiple projects. In particular, concurrent project management increases the challenges associated with resource optimization and planning. In multiple projects, project deliverables are viewed as an integrated set of portfolio activities (Dooley et al. 2005). As such, the main responsibility of managers in multiple projects is to ensure control and to balance resources optimally. There are four key challenges that project managers face in multiple projects. The multiple project activities overlap is one of the main challenges. In such a case, project activities may even overlap with day-to-day company operations, sharing of company resources such as equipment and devices across different projects, resource prioritization between projects based on their weights and modes, and meeting key project deadlines.

Like in single projects, another key problem affecting multiple projects is the change of scope. Change of scope occurs because of project alteration or expansion.

The change of scope may also occur due to unexpected circumstances that need additional resources. Managing change of scope, especially for multiple projects, requires flexibility in decision-making and the optimization of resources (Rieck and J. Zimmermann, 2015).

In a study by Dooley et al. (2005), the main sources of failures in the management of multiple projects are poor leadership, alignment of project goals and project objectives, control, planning, and monitoring. Another key aspect that contributes to the failure of multiple projects is that organizations do not learn from past project mistakes. By learning from past mistakes organizations can incorporate past knowledge into existing project models (Dooley et al. 2005). The reuse of knowledge learned from past mistakes can significantly improve decision-making and problem-saving in existing project models.

A critical aspect in multiple project management is resource planning and management. It facilitates the efficient use and reuse of resources in a project. For instance, in construction projects, the main resources are known as “the 3Ms”. These include people, materials, and machinery. In resource planning and management, the key objective is usually to prevent resource overload using three critical objective functions (Rieck and J. Zimmermann, 2015). These functions include minimization of minimum moments, maximum overload cost issues. This relates to the costs that are generated when the use of a certain resource is exceeded, and the total adjustment cost issue that occurs where there is a need to minimize costs that are caused by the increase in resource utilization.

1.3 Resource Management

Resource management is a crucial aspect of project management, especially for resource-constrained projects. In multiple project management, resources may be limited by the simultaneous occurrence of two or more resource-intensive activities. Resource planning and management techniques may therefore be useful to manage resource usage and minimize resource variations. For any project, resource management can be classified into three primary types: resource scheduling, resource leveling, and resource allocation. Each of these categories can be modeled to ensure optimal resource utilization.

Resource scheduling gives an overview of the utilization of resources in the period of the project. Most researchers refer to resource scheduling as resource loading (Project Management Institute, 2013). Resource scheduling is a widely used resource management technique and is often used by project planners. Today, most projects utilize computerized scheduling techniques which provide the ability to seamlessly organize and present project information. resource scheduling can also be accomplished using a network model process.

Resource leveling is concerned with addressing the peaks and valleys in the management of resources without increasing the duration of the project. resource leveling is accomplished by redistributing the start and finish activities using the float time of non-critical activities. The goal of resource leveling is to reduce or eliminate resource conflicts (Rieck and J. Zimmermann, 2015). Since project leveling only interferes with noncritical activities, both the project duration and the critical remain untouched. Resource leveling is often used when there is a fixed project completion date.

Resource allocation also plays a key aspect in resource management. A proper understanding of resource allocation is critical in the design of resource optimization models. Resources allocation techniques set maximum limits for the specific project activities according to existing heuristic models. The main objective of these techniques is to schedule activities in a way that does not exceed existing resource limitations. In this context, projects are finished in very short times.

In resource management, scheduling and planning are closely connected. As such, neither of the problems can be solved in isolation. Both scheduling and planning affect the decision-making of companies. For instance, in a factory setting, the relationship between scheduling and planning must be considered because it influences several aspects of decision-making such as the input factors, the inputs that require optimization, the type of scheduling challenges, and the objective of the solution to the problem. However, resource planning and scheduling may have differences that depend on the level of differences in the solution models (Hariga and El-Sayegh, 2011). Other sources of differences are the objective of the problem and the scheduled time.

Despite the differences, scheduling models are widely used in resource planning. The models. In factories, scheduling methods are used to determine the ideal production level and the storage levels that can satisfy a given level of demand given the cost. Unlike scheduling models, however, planning models usually use aggregate models. Aggregate models are required to subject specification to different costs. Planning models may also be large because of the time-periods involved. The solutions of a planning model can be used as an input to a scheduling problem. Aspects of planning such as batching decisions determine the type of decision required at the scheduling level.

Another key association between scheduling and planning is the flow of information between them. In particular, an optimized schedule facilitates proper control referencing. As will be illustrated in the below sections, researchers have historically approached scheduling as an isolated problem. Today, most experts agree that scheduling incorporates control and planning. Literature, in this area, considers scheduling and planning as an integrated set of problems.

Scheduling and planning problems like “Resource-Constrained Project Scheduling Problem (RCPSP)” and “Multiple Resource-Constrained Project Scheduling Problem (MRCPSP)” are also subject to a wide range of constraints and characteristics. Although these factors increase the complexities of the problems, they are needed to increase the feasibility of solutions. Resource constraints may include utilities, auxiliary units, and human resources. Resources can either be renewable or non-renewable. The renewable category of resources can either be continuous or discrete. The different classification shows the level of complexity that is associated with scheduling problems and the diversity required in handling when designing industrial applications. Although the current solution proposes a mixed integer model, there exist other approaches such as metaheuristics, heuristics, and constraint programming.

1.4 Problem Motivation

Our main motivation to work on this problem comes from real issue in IT division at the Ministry of Administrative Development, Labor and Social Affairs (ADLSA) in Qatar. Project managers facing this issue in management of multiple projects. There are a challenge in everyday management, how can project managers charge and developing software or applications, for different ministries or different departments. The ministry has many resources with different qualifications such as developers, programmers, analysts, and projects managers. They have many projects, these projects are independent, each project have specific due date. The main problem that projects manager may face is how to allocate these resources within a given time horizon so that they can complete the projects without having tardy tasks. How can projects manager meet the requirements, is there a need to add extra capacity or need more staff to complete all tasks without delaying the tasks to get the best allocation that will lead to reduce the cost and time.

CHAPTER 2: LITERATURE REVIEWS

2.1 Literature Reviews

The current review investigates the existing optimization models for multiple resource planning. The challenge of achieving optimal scheduling and resource planning has been a subject of significant scholarly interest. To date, a wide variety of factors has been used to attain optimization. These include dynamic programming, neural networks, expert systems, linear programming, and genetic algorithms (Dooley et al. 2005). The review starts with an overview of the challenges that managers encounter during multiple resource planning. The review then analyzes extant literature for optimization and resource planning solutions for multiple projects.

2.2 Optimization Models

Optimization is mainly connected with the use of scientific models to determine the best course of action during resource management. In recent years, optimization models have become crucial tools in ensuring resource optimization and the economic feasibility of projects. In particular, optimization models are widely applied in resource planning, decision-making, and scheduling. With the advent of technology and computing, sophisticated optimization models have been developed to solve the wide range of issues associated with resource planning, scheduling, and optimization. The four main categories of optimization models are analog methods, analytical methods, heuristic methods, and metaheuristics algorithm methods.

Analog methods utilize dual series, electrical methods, and physical planning methods. Analog methods are complex to design and are usually not reliable. Analytical methods use mathematical methods to schedule different project tasks. However, the methods are ineffective when used in large-scale projects (Project Management Institute, 2013). Heuristic methods use computers to perform equations that facilitate project planning and scheduling. Lastly, genetic algorithms are based on genetic science.

Some of the earliest used algorithms in project planning were evolutionary algorithms (EA). EA algorithms are designed to mimic natural processes. These algorithms have been used in the optimization of critical project areas such as resource scheduling, time-cost trade-off, project control, risk prediction, finance-based scheduling, cost, and duration estimation, logistic operations, equipment selection, scheduling, and financial administration.

2.2.1 Existing Literature in Optimization Models

A wide range of optimization models exists in the literature. One of the earliest project planning and optimization models was a simple heuristic algorithm called "the Minimum Moment algorithm". The primary objective of the model was resource leveling and optimization. The model was proposed by Harris (1973). It was later improved by Mohammed (2000) who took into consideration the free float factor of activities and the selection criteria of resources. The model also minimized the deviations between actual and optimal resource utilization. In a different study, Ramlogan and Goulter (1998) designed a model to enable project scheduling and resource-leveling. The model has three main objectives: optimal resource allocation, internal resource allocation, and resource duration minimization. The researchers used a mixed-integer model and a weighted multi-objective algorithm.

Several researchers have proposed meta-heuristic models for scheduling and resource planning. A research by Senouci and Eldin (2004) proposed a metaheuristic based on a "genetic algorithm". The model concurrently performs resource allocation and leveling. In a much recent study, Liao et.al (2011) provides a study of several metaheuristics approaches to optimization.

The research then proposes a generic algorithm to minimize deviations in resource scheduling and planning. In a different study, Hariga and El-Sayegh (2011) designed a meta-heuristic optimization model for multi-resource leveling problems.

In summary, the main exact optimization models existing in the literature include zero-one integer programming, dynamic programming, and implicit enumeration (enumeration with branch bound). On the other hand, the main heuristic models are multi-pass and single-pass methods. Tabu search, genetic algorithms, and simulated annealing are the main metaheuristic approaches that have been used to solve MRCPSP. Other heuristic solutions are population-based approaches, local search-oriented approaches, neural networks, and forward-backward improvement. These methods are widely used in different areas of multiple resource planning and management. The methods are also associated with different sets of advantages and disadvantages. Some are only suitable for small scale, while others can be applied in complex projects.

To solve MRCPSP problems, dynamic programming solutions divide problems into sub-problems. After every small problem is solved, the program combines the solutions to solve the whole problem. One of the first programs to solve an RCPSP problem was developed by Carruthers and Battersby (1966). The program solved the problem by finding the maximum path using the problem symmetry of the network. Although the method was effective in solving a network problem, it could not be used to solve practical problems.

A wide variety of research has been carried in an attempt to use zero-one integer programming to solve MRCPSP problems. These include studies by Patterson and Roth (1976) and Patterson and Huber (1974). In particular, there exist several programming solutions to shop scheduling problems. A program developed by Patterson and Roth (1976) performed linear programming as a solution to MRCPSP problems.

In particular, the study used a zero-one variable to distinguish between the start time and finish time of multiple projects. On the other hand, Patterson and Huber (1974) produced minimum duration schedules by using both bounding techniques and zero-one programming.

As noted by the studies conducted by Patterson and Huber (1974) the number of variables in the problem size in zero-integer programming increases as the problem size increases. As a result, the programming is ineffective when solving complex problems. Zero integer programming can therefore only be applied in simple or small-size problems. It is ineffective in practical solutions. However, the structure of the zero-integer algorithm has significantly reduced the computational efficiency in solving MRCPSP problems. In particular, the program introduced implicit enumeration algorithms that reduced the computational time for solving MRCPSP problems.

Branch and bound approaches have been widely used in solving implicit enumeration issues. In these solutions, the model schedules and delays activities based on specific precedence and resource constraints.

The number of schedules created in the model is dependent on the number of combinations. The program is designed to create as many partial schedules as the number of feasible combinations. Implicit enumeration with branch and bound have been widely used in solving MRCPSP problems. These include solutions developed by researchers like Fisher (1973), Hastings (1972), and Christofides et al. (1987).

In a branch and bound solution given by Davis and Heidorn (1971), the researchers find the solution to an MRCPSP problem by transforming the problem into a problem that seeks to find the shortest path in a graph. In such a setting, the solution can be determined by scheduling tasks in a given period. The only drawback of the model is that it is only applicable to easy problems and ineffective in solving complex problems.

Although they are widely applied, combinatorial approaches are known to have several technical drawbacks. One of the primary drawbacks is that the models are only effective in simple data sets. Very few models for solving complex data sets exist. Secondly, existing solutions to multiple resource planning using combinatorial are ineffective in solving both in terms of computational efficiency and solution quality. The problems cannot provide effective solutions to complex practical problems with hundreds or thousands of multiple project activities. These approaches are therefore only applicable in small problems with a limited number of project activities. Some researchers have also used "Lagrangian relaxation of resource constraints" to create lower bounds. Lagrangian relaxation is a type of "linear programming relaxation" (Fisher 1973).

One of the most effective exact approaches was developed by Demeulemeester and Herroelen (1992). The algorithm developed by the researchers performed well compared to all other algorithms. Unlike the other algorithms, the one by Demeulemeester and Herroelen (1992) is based on a technique called a "depth-first solution". In this technique, the nodes in the solution tree are representative of the resource and partial schedules. On the other hand, branches represent different combinations of activities. The computational results of the algorithms developed by the researchers showed that they performed better when compared to other similar algorithms. In a different branch and bound algorithm, Brucker et al. (1998) used disjunctive constraints between a set of activities.

In a different study conducted by Mingozzi et al. (1998), the researchers designed a zero to one linear program for MRCPSP problems. The main objective of the program was to derive lower bounds with the capability of showing the most optimal path in the graph. The developed algorithm showed that it could solve hard instances that other algorithms were incapable of solving.

In a study by Dorndorf et al. (2000), the researchers developed a "branch-and-bound algorithm” that computes the start and end times of activities. The algorithm uses a “constraint-propagation technique” to reduce search space.

Priority-rule-based heuristic approaches have also been widely used to solve MRCPSP problems. These approaches can broadly be categorized in two: single-pass and multi-pass approach. “Multi-pass methods” can further be divided into three: “sampling methods, forward-backward scheduling methods, and multi-priority rule methods”. Heuristic approaches make use of different priority rules to generate schedules. Serial Generation Scheme (SGS) are used to produce multiple schedules. In an SGS methodology, feasible schedules use priority ranking and are built up in a stepwise design. The two types of SGS are the serial SGS and the parallel SGS. Serial SGS works as an activity-oriented scheme, whereas parallel SGS is a time-driven scheme. Both types of SGS can be used to decode schedule representation.

In a “single pass method”, only one schedule is used. Some examples of priority rules used in the single-pass approach are the earliest start time rule, the latest finish time rule, and the most total successor rule. “Single-pass method” has been widely applied to RCPSP problems.

A multi-pass method uses more than one schedule, each iteration is associated with a different priority rule. The process is repeated several times until the optimal solution is obtained. Several studies have used multi-priority rules to solve RCPSPs. In a multi-pass method developed by Ulusoy and Özdamar (1989), the researchers use a Weighted Resource Utilization and Precedence (WRUP) rule. The study compares the rule to other rules such as Latest Finish Time (LFT) and Minimum Slack (MINSLK). The study then establishes relations between the resource characteristics of RCPSPs and heuristic techniques.

The results of the study showed that WRUP could solve problems at a higher success rate than other types of heuristics. A multi-pass methodology was also used by Boctor (1996). The study assessed several heuristic rules to investigate the relationship between the obtained solutions and the number of heuristic rules applied. The study showed that a combination of four or five heuristic rules can yield an optimal solution for large RCPSP problems. The methods can be broadly classified into two: forward back scheduling, multi priority rule, and sampling methods.

MRCPSP problems have also been solved using forward-backward scheduling methods. In these methods, an SGS is used to iteratively perform a forward and backward schedule.

Sampling methods have also been widely applied in solving MRCPSP problems. Sampling methods use a combination of priority rules and SGS. Unlike forward back scheduling methods, different schedules can be obtained from sampling methods. The most optimal schedule is selected from the options. In a sampling technique developed by Cooper (1976), a randomized technique is used to select the best schedule. Studies that have used sampling methods have shown that the methods have higher computational efficiency compared to other deterministic and heuristic methods.

As discussed earlier, the main types of metaheuristic approaches that have been applied to solve MRCPSP problems are tabu search, simulated annealing, ant colonies, and genetic algorithms. Since genetic algorithms were introduced by Holland (1975), they have been widely used to solve MRCPSP problems. Today, genetic algorithms are used both as an optimization technique and as a learning and adaptation model. In an algorithm developed by Hartmann (1998), it was found that the results were better than those produced by other simulated annealing techniques.

Simulated annealing techniques are used to solve complex combinatorial problems. The technique through a search and improvement method. The basic concept of the format is that it starts with a feasible solution then the algorithm periodically improves the solution until no improvement is required. Researchers like Valls et al. (2005) and Boctor (1996) have widely used the simulated annealing approach. In the study by Boctor (1996), a non-preemptive technique for solving an MRCPSP problem is proposed. The method is renewed from time to time. The method was found to have a higher level of efficiency when compared to the tabu search method.

The study conducted by Bouleimen and Lecocq (2003) proposed a solution to an MRCPSP problem using simulated annealing. The method used both SGS schedules and an activity list representation to solve the MRCPSP and RCPSP problems. To solve the RCPSP problem, a time-increasing process and an alternated activity process was used. For the MRCPSP problem, the researchers used a mode search neighborhood and a double embedded search loop. The study proved the algorithm has a higher level of efficiency. In a simulated annealing study by Valls et al. (2005), it was shown that the technique improved computational efficiency.

Tabu search has also been used to solve MRCPSP and RCPSP problems. The method utilizes a combination of a heuristic and a "meta-heuristic method". The meta-heuristic was superimposed on the heuristic model. The technique works by avoiding cycle entrainment and penalizing moves in different sets of iterations. The first step a tabu search algorithm takes is matching the search to a local minimum. The search then records moves in a tabu list to prevent any retracing of moves. The list is stored in a tabu search record. The search algorithm was also studied by Nonobe and Ibaraki (2002) as a solution to an RCPSP problem. The researchers use an activity list representation, a serial SGS, and a neighborhood reduction mechanism.

The SGS also contained features such as the availability of renewable and non-renewable resources and multi-mode processing.

In an ant colonies approach, a meta-heuristic approach is used to solve MRCPSP and RCPSP problems. Using an ant colonies approach, Dorigo et al. (1996) applied “the classical Traveling Salesman Problem (TSP)” technique. The solutions were found to have a higher level of computational efficiency. The primary features of the system were the use of heuristic procedures, distributed computations, and positive feedback. The result of the experiment showed that the ant colonies approach was robust and effective. In a different research, Merkle et al. (2002) used an ant colony approach to solve an optimization problem. In the study, the authors combine heuristic and ant colony algorithms. In particular, the method is combined with the paper algorithm developed by Hartmann and Kolisch (2000). The combination yielded an algorithm with an efficiency level that was higher than that provided by the other types of algorithms (sampling method, tabu search, and simulated annealing).

Another approach that has been used to solve optimization problems is the local search-oriented approach. Unlike other methods, the local search-oriented approach does not rely on metaheuristic techniques. Researchers like Valls et al. (2005) and Palpant et al. (2004) have attempted to use the local search-oriented approach. The study by Valls et al. (2005) utilized a double phase algorithm that is based on the serial SGS and a topological order representation. The method was found to result in a higher level of computational efficiency.

Neural network approaches have also been used to solve scheduling and resource allocation problems. In a study by Colak et al. (2005), the authors propose the use of a neural network algorithm to solve an MRCPSP problem. The algorithm uses a combination of an SGS based augmented neural network and a serial SGS.

The algorithm also uses a forward-back improvement technique in a hybrid approach.

The results of the study showed that the algorithm produced a good performance compared to heuristic and deterministic approaches. The drawback of neural networks is that they require a high level of training. These algorithms can therefore be classified as trial and error algorithms.

2.3 Optimization Solutions for Multiple Resource Planning

As illustrated earlier, the problem of research planning and scheduling has always existed in project management. The problems are higher in multiple projects compared to single projects. Most of the existing research in resource optimization focuses on the use of heuristic methods. Today optimization most models also utilize Genetic algorithms (GA). These algorithms are based on natural selection processes and can be used for constrained and unconstrained optimization problems (Senouci and Eldin, 2004). The algorithms copy the natural process of survival of the fitness and other behavior of species. The metaheuristic developed by this algorithm can solve optimization problems.

Genetic algorithm is a group of algorithms that model solutions to optimization problems using a technique inspired by the process of evolution. In particular, genetic algorithms encode optimization solutions in a way that mimics chromosome data structures. Genetic algorithms act as function optimizers (Liao et.al, 2011). The algorithms periodically modify a set of solutions to give the most optimal solution. Genetic algorithms can particularly be used to solve problems with an objective function that is stochastic, non-differentiable, nonlinear, or discontinuous.

The main components of a genetic algorithm are the cost, optimization function, and optimization variables. Solutions to GA functions are computed using computer simulations in which sets of abstract representations to optimization problems are evolved periodically to give better solutions (Senouci and Eldin, 2004). The solutions are expressed in the form of binary strings of 1s and 0s. Genetic algorithms first select solutions randomly from a population. The solutions are then modified and evaluated periodically and then used to form a new population. The algorithm arrives at an optimal solution when the maximum number of evolutions is attained.

The key terms used in GA are fitness function, individuals, generations and populations, and encoding. The fitness function is the function that the algorithm attempts to optimize. In project management, the fitness functions may be designed to solve scheduling or resource planning problems. Individuals are the point where fitness functions are applied. The individual is the single solution to the computed fitness problem. The solution which the algorithm is attempting to solve is designed using chromosome parameters (Hariga and El-Sayegh, 2011). Strings are used to represent chromosomes.

The term population is used to describe an array of individual solutions. With every iteration, the algorithm performs a series of computations on the population to produce children (a new set of the population). The algorithm selects the population that shows high levels of fitness. Every newly generated population is called children (Senouci and Eldin, 2004). Encoding is the process by which a solution is represented in the form of a string. The string conveys the requisite details. The algorithm operates in the same way genes reveal the character of a person. Each part of the solution is represented by a bit in the algorithm. Value and binary encoding are both used in GAs.

2.3.1 Resource-Constrained Project Scheduling Problem (RCPSP)

In constrained resource planning, the primary objective of the problem is usually to ensure efficient resource utilization. Such models can be modeled using combinatorial auction. During the scheduling and distribution of resources, project managers of different projects act like bidders competing for distributed resources. Each project in multi-resource planning has a set of unique requirements such as (resource constraint, the capacity of resources, and activities that project managers have to deal with). This may include different lead times and different project processing requirements. Apart from efficiently distributing resources, the model can allow project managers of different projects to request a complex combination of resources. In multiple resource planning, such combinations are referred to as multisets (Wellman et al., 2001).

Constrained-resource problems were in the early years solved using mathematical models like a branch and bound, linear programming, integer programming, and dynamic programming. Existing project scheduling solutions make significant use of heuristic rules. Heuristics are widely used because of their simplicity and efficiency. However, they do not always result in optimal solutions. Mathematical solution is efficient on a small scale but inefficient in large scale complex problems.

Today, modeling solutions as Resource-Constrained Project Scheduling Problem (RCPSP) are applied in a wide range of business solutions. These include cloud computing workflow scheduling, software development, and manufacturing. RCPSP mainly aims at finding the optimal start time of a resource-dependent activity and optimizing performance in a way that resource constraints are respected.

RCPSPs are centralized and deterministic problems. As such, there exist available information regarding the problem and a single decision-maker.

However, several assumptions have been developed to adopt the problems to respond to the dynamic nature of today’s business world. Two primary factors are relevant in the development of a dynamic-solutions to existing real-world problems. These are distributed management and execution uncertainty. By factoring in distributed management, existing models have to incorporate solutions that adhere to both privacy and distributed decision-making. Execution uncertainty is also a critical factor in the development of effective optimization and planning models. In particular, the uncertainties experienced in multiple projects need to be incorporated into the models. All the existing mechanisms for solving RCPSPs are broadly classified into two: heuristic approaches and exact approaches. Exact approaches are those approaches whose effectiveness and reliability have been statistically proven. On the other hand, heuristic approaches are those that use computational techniques to find solutions to problems. Most approached mainly concentrate on the development of Mixed Integer Linear Programming (MILP) approaches to solve RCPSPs. After the development of the MILP, solvers like CPLEX and Gurobi can then be used.

A wide range of MILP formulations is available in the literature. These include event-based formulations, continuous-time formulations, and discrete formulations. Although the models are known to be effective, studies have shown they are not scalable (Brucker and Knust, 2012). Models that utilize exact approaches may be designed using constraint programming techniques. Such a design finds the optimal schedule through a combination of backtracking search and constraint propagation mechanisms. The key techniques used in RCPSP problems are energetic reasoning, timetabling, lazy clause generation, and edge fitting (Schutt et al., 2013).

According to most studies, constraint programming-based approaches have a higher level of computational efficiency compared to MILP based approaches.

This is mainly because of the reduction of search spaces by the active exploration of constraints (Schutt et al., 2013). Most exact approaches run in exponential time to arrive at the best solution. Since they are anytime algorithms, they can be terminated early and still be able to provide the most optimal solution.

Heuristic approaches to solving RCPSP have also been widely studied. Perhaps the most common approaches are metaheuristic approaches and schedule generation schemes. Scheduled generation schemes perform operations using a set of priority rules. They are thus simple and flexible. Because of these properties, scheduled generation schemes are widely used in building solutions for RCPSPs (De Nijs and Klos, 2014). On the other hand, metaheuristic techniques employ random techniques in the design of searching algorithms. In general, metaheuristic-based algorithms require less computational time compared to algorithms that utilize exact approaches.

Existing research also shows the use of combinatorial auction-based approaches to solve scheduling problems in resource planning and optimization. Combinatorial auction-based approaches for multiple resource planning have been used to solve RCPSPs. The approach uses the Lagrangian decomposition in generating solutions to problem combinations. However, the use of combinatorial auction-based approaches is known to result in infeasible solutions and schedules (Wellman et al., 2001). Most scholarly study has also gone into the development of planning and optimization techniques for uncertain situations. Such solutions incorporate MILPs for deterministic RCPSPs. The constraints ensure a low level of planning and schedule violation (Varakantham et al., 2016).

2.3.1.1. Multimode Resource-Constrained Project Scheduling Problem (MRCPSP)

Non-preemptive execution techniques are the main techniques used in Multimode “Resource-Constrained Project scheduling problems (MRCPSP)”.

These problems mimic the challenges experienced in solving multiple resource optimization and planning problems (Varakantham et al., 2016). In particular, every mode of execution has a set of execution requirements and a prescribed duration. The resource requirement may be renewable or non-renewable. A wide range of solutions to solve MRCPSPs have been proposed in the literature. These include simulated annealing, heuristics, and serial scheduling schemes.

As an extension of the RCPSP, MRCPSP is concerned with the determination of optimal scheduling in instances of shared resources. In MRCPSP, the duration of each task is represented as a function of the resources and level of the resources used. As a solution to multi-project planning, MRCPSP has been applied in scheduling and resource optimization (Varakantham et al., 2016). Notably, MRCPSP is more complex compare to RCPSP. The MRCPSP problem becomes non-deterministic polynomial-time hard (NP-hard) when there are two or more resources are nonrenewable. The complexity of MRCPSP is further increased in the instance where the model allows for the choice of modes.

Relevant to the existing review, optimization solutions for multiple resource planning can be organized in a set of precedence of activity sequence. To create a valid sequence, researchers have used the Variable Neighborhood Search (VNS). In particular, a VNS technique is associated with the exploration of neighborhood structures used in search steps that generate an optimal solution. VNS based heuristic approaches significantly increase the probability of obtaining the most optimal solution through random selection. In a study by Chakrabortty, Abbasi, and Ryan (2019), the researchers found a near-optimal solution for a multi-mode resource-constrained scheduling problem.

One of the first solutions to MRCPSP was developed by Boctor (1996).

The first solution developed by the researcher was a heuristic single-pass approach that utilized a parallel scheduling scheme. In the model, activities are defined by the decision set of the predecessors. In particular, the MLSK priority rule defines the decision set. The mode with the shortest decision time defines the scheduled activity. For simulated annealing, the algorithm is iterative and keeps repeating the solution until an optimal solution is obtained.

In the simulated annealing (SA) algorithm, the solutions are represented in a list form where a solution's position represents its level of priority. Activities are then chosen at random to result in a neighbor solution. In addition, the Shortest Feasible Mode (SFM) rule is used to select the ideal activity mode. In a different study, Drexl and Grunewal (1993) use a random sampling approach that utilizes a serial scheduling scheme with an SPT priority rule. The ideal time of selected activities is then determined in consideration of the existing constraints. The research resulted in a model solution with a deviation of 3.5% from the optimal solution.

In research conducted by Hartman (2001), an MRCPSP problem is solved using a genetic algorithm. The algorithm relies on a set of feasible activities and a combination of different modes. The model also utilized a serial scheduling scheme to generate a schedule. The researchers found that the genetic algorithm resulted in a better result compared to the solution by periodic rule encoding. Some researchers have also proposed a genetic-based local search algorithm. The first phase of the algorithm performs a global search, whereas the second phase does a local search. The global search collects elite solutions which then form the population of the second search.

2.3.2. Resource Loading Problem (RLP)

In a study by Hans (2001), the author studies RLP by factoring precedence constraints and allowing pre-emption. After studying an RLP problem, a study by Kis (2005) proposes a branch and cut algorithm. The authors describe an RLP problem as a project scheduling task with high-intensity activities. To solve an RLP problem, Gademann and Schutten (2005) propose a linear programming heuristic technique. In a study by Wullink et al. (2004), the researchers propose a scenario-based approach to solve an RLP problem. Unlike the study by Kis (2005) and Hans (2001), the study by Song et al. (2019) proposes minimum intensities for order execution. With minimum intensities, an advanced linear description of the feasible intensity is attained. The study by Song et al. (2019) concludes that the branch-and-cut algorithm has a higher level of efficiency.

RLP problems have also been investigated by researchers like Blazewicz et al. (2004) where tasks are executed by several processors. In such a setting, task processing is represented by a non-linear function of its allocated processors. In a study by Nattaf et al. (2019), the researchers propose a resource scheduling solution for a resource scheduling problem that aims to minimize resource consumption. A study by Fundeling and Trautmann (2010) also solves a project scheduling problem. In particular, the study investigates the minimum and maximum level of resource usage in the completion of a project. In a different study, Naber and Kolisch (2014) provide a solution for an RCPSP problem. Notably, the RCPSP solved by the researchers utilizes a flexible resource profile. The authors use different MLP variations to solve the problem.

Time-indexed formulation has also been widely used to solve scheduling problems that have fixed processing times. In a study by Sousa (1992), a single machine schedule is performed using a time-indexed formulation.

The model was based on the strength of the linear programming relaxation. To solve

RCPSPs, some researchers have proposed several polyhedral solutions that have been provided by researchers like Bianco and Caramia (2017) and Artigues (2019). Research by Naber and Kolisch (2014) and Burgelman and Vanhoucke (2018) also provides computational results for time-indexed formulations.

As illustrated by Song et al. (2019) resource planning and scheduling can also be accomplished through the use of a branch-and-cut algorithm to solve a resource loading problem (RLP). RLP problems can be designed to mimic the resource planning challenges experienced in multiple projects. The development of RLP problems has historically been driven by real-world problems. For instance, a study by De Boer (1998) attempted to resolve employee scheduling and planning at a real-world organization. In a different study, Belien et al. (2012) used an RLP problem to design scheduling and workplace plans for the resources and equipment in an aircraft company.

2.2.2.1 Resource Tardiness

There have also been studies that have addressed the problem of Resource Tardiness Weighted Cost Minimization in Project Scheduling. A study by Shirzadeh Chaleshtari (2017) analyzed the problem of maximization under the tardiness penalty costs. The study uses a CPLEX solver-based algorithm and makes use of the original RCPSP problem. CPLEX was compared to a branch and bound algorithm. The branch and bound algorithms were found to have a higher level of efficiency. The study showed a higher level of algorithm efficiency even at different levels of difficulty.

2.4 Summary

In the literature review, we started with existing literature in optimization models their strengths and limitations, then we describe the main resource loading problem.

After that, we presented a brief description of the resource-constrained project scheduling problem (RCPSP) and some studies related to the problem. Then, we described the multimode resource-constrained project scheduling problem (MRCPSP) and the most important researches related to the problem.

Chapter 3 will present the proposed model of optimization models for multiple resource planning and the different enhancement procedures to fill the gap in the literature.

2.3. Summary of different multi-resource planning models

The below table gives a summary of the features of different multi-resource planning and optimization models.

Table 1. Summary of the Literature Review and Comparison of Reviewed Models

Reviewed Author	Year	Method	Datasets
1- Boctor	1996	Heuristic	Own
2- Chakrabortty, R., Abbasi, A., & Ryan, M.	2019	Heuristic	Own
3- Drexl A. and J. Grunewald	1993	Heuristic	Own
4- Dooley, Lupton, and D. O'Sullivan	2005	Portfolio management	Case study
5- Hartmann	2001	Generic Algorithm	Project Scheduling Problem Library
6- Hariga and S. M. El-Sayegh	2011	Mixed binary linear optimization model	Own
7- Liao	2011	Metaheuristics	Own
8- Mohammed A. Salem Hiyassat	2000	Modification of minimum approach	Own
9- Ramlogan, R., and I. C. Goulter	1989	Mixed Integer Model	Own
10- Senouci A.B and N. N. Eldin	2004	Generic Algorithm	Own
11- Shirzadeh Chaleshtari, A.	2017	CPLEX & a branch and bound algorithm	Project Scheduling Problem Library
12- Carruthers and Battersby	1966	Critical path method	Own
13- Patterson and Roth	1976	Zero-one integer programming	Own
14- Patterson and Huber	1974	Zero-one integer programming	Own
15- Fisher	1973	A branch-and-bound algorithm	Own
16- Christofides et al.	1987	A branch-and-bound algorithm	Own
17- Davis and Heidorn	1971	A branch-and-bound algorithm	Own
18- Demeulemeester and Herroelen	1992	A depth-first solution	Own
19- Brucker et al.	1998	A depth-first solution	Own
20- Mingozzi et al.	1998	A zero to one linear program	Own
21- Dorndorf et al.	2000	A branch-and-bound algorithm	Own
22- Ulusoy and Özdamar	1989	Weighted resource utilization and precedence (WRUP)	Own
23- Cooper	1976	A combination of priority rules and SGS	Own
24- Holland	1975	Genetic algorithms	Own
25- Valls et al.	2005	Simulated Annealing	Own
26- Bouleimen and Lecocq	2003	Simulated annealing	Own
27- Nonobe and Ibaraki	2002	Tabu Search	Own
28- Dorigo et al.	1996	Ant colonies approach	Own
29- Merkle et al.	2002	Ant colony approach	Own
30- Palpant et al.	2004	A local search-oriented approach	Own
31- Colak et al.	2005	Neural networks	Own
32- Kis	2005	A branch and cut algorithm	Own
33- Gademann and Schutten	2005	A linear programming heuristic technique	Own
34- Sousa	1992	A time-indexed formulation	Own
35- Bianco and Caramia	2017	Polyhedral solution	Own
36- Naber and Kolisch	2014	Time indexed formulations	Own
37- Burgelman and Vanhoucke	2018	Time indexed formulations	Own

CHAPTER 3: A NOVEL MODEL AND SOLUTION FOR THE MULTIPLE RESOURCE LOADING PROBLEM

The current chapter gives an in-depth overview and solution for the MRLP problem. A novel model and solution for the MRLP problem are presented. The model provides a solution that minimizes time wastages and ensures efficient resource utilization in resource-constrained planning. Apart from facilitating efficient allocation of scarce resources, the model allows the complex combination of resources while taking into account restrictions and time constraints.

The chapter is divided into four distinct parts: section 3.1, section 3.2, section 3.3, and section 3.4. Section 3.1 gives a detailed problem definition that prompts the development of the novel model and solution for the MRLP. Section 3.2 introduces a mixed-integer programming (MIP) formulation of the outlined problem. The third section 3.3 describes the key model enhancement of the outlined problem. Lastly, Section 3.4 introduces a MIP-based decomposition heuristic. The decomposition heuristic is formulated to provide an efficient solution for large-scale instances.

3.1 Problem Definition

The current research provides a general solution for a multiple resource planning and scheduling problem. MRLP problems arise in almost every organization undertaking multiple projects. A lot of organizations today face multiple resource planning project complexities, especially those in the construction, engineering, and manufacturing industries. The key objective of multiple resource planning and the formulation of MRLP problems is to ensure projects adhere to the key constraints of the project's performance, time, and budget.

3.1.1 Approach/Methodology

The MRLP seeks to create a solution that solves the four key challenges that project managers encounter in day to day management of multiple projects. In particular, the objective is to enhance project scheduling within a given time horizon to get the optimal resource allocation solution that minimizes the cost of tardiness and the cost of additional capacity if any.

Multiple project management involves balancing competing project interests. The main responsibility of managers in such settings is to ensure multiple project objectives are met. The key issues that may arise in the management of multiple projects are overlap of activities and tasks, resource sharing, competing project deadlines, and resource prioritization.

In multiple resource planning, resource allocation is often a problem because of time criticality, dealing with high project demands, uncertainties, project constraints, and dealing with competing priorities. The current problems focus on renewable resources such as employees and machines. Such resources need optimal resource allocation that minimizes the cost of tardiness and the cost of additional capacity. Failure to efficiently allocate resources has been the main cause of project delays in multiple resource planning.

3.1.2 Modelling Problem Formulation

A Multiple Resource Loading problems (MRLP) problem is formulated as a mixed-integer linear programming (MILP) problem. The problem formulated in this chapter can generally be applied in any multiple resource planning problem. The primary objective of the model formulation is to develop an MRLP that minimizes the total cost of tardiness and the total cost of additional capacity.

For each project, number of tasks, number of available resources, number of execution modes of each task, consumption needed of each resource by different task under each mode, starting date, duration, weight, cost of adding extra units of capacity to the resource and the deadline is given.

Notably, the project is uploaded based on the processing time in which the resource should be available at the time. Available resources can be identified based on capacity. The objective functions and the constraint conditions can be achieved by integrating constraints with decision variables.

3.2 Problem Formulation

3.2.1 Notation

For Multiple Resource Loading problems (MRLP), the number of tasks can be represented as the set n number of tasks to be executed within H time frame. The H is represented as a scheduling time horizon and is considered discrete for periods that are similar in length. For the current model, the period (months) is represented as t. The time t spans over a specific time interval represented as $[t-1, t]$ for t = 1… H. The amount of resources in the project is represented as r $(r=1\ldots R)$. The resources exist such that there is no time t where the resources exceed the availability of renewable resources R. Within each time t, there is a specific capacity b_{rt} of the main resource r. Moreover, each task j $(j = 1\ldots n)$ is executed under m_j number of execution modes. Each task j $(j = 1\ldots n)$ has a consumption a_{jrk} of resource r by task j under mode k. Each task j $(j = 1\ldots n)$ has a processing time p_{jk} under mode k. Each task j $(j = 1\ldots n)$ has a real start date r_j of the project. The due date of task j is given as d_j. Each activity j is to be completed within a duration p_{jk} and in the utilization of a specified resource r. As such each task has a specified duration and a set of resources, tasks are also assigned different weights w_j which denotes task significance/priority and cost of adding capacity σ_{rt}. Specifically, σ_{rt} is the cost of adding one unit of resource r at time t.

In summary, the following is a list of input data and notation utilized in the study:

n: Number of tasks,

R: Number of resources,

H: Time horizon,

b_{rt}: Capacity of resource r at period t,

m_j: Number of execution modes of task j,

a_{jrk}: Consumption of resource r by task j under mode k,

p_j: Processing time of task j under mode k,

r_j : start date of the project

d_j: Due date of task j,

w_j: Weight of task j,

σ_{rt}: Cost of adding one unit of capacity to resource r at period t.

3.2.2 Decision Variables

We define the following decision variables:

x_{jk}: Binary variable that takes value 1 if task j is executed under mode k, and 0 otherwise.

y_{jt}: Binary variable that takes value 1 if task j is executed during period t, and 0 otherwise.

s_{jt}: Binary variable that takes value 1 if task j starts at the beginning of period t, and 0 otherwise (that means, $s_{jt} = 1 \Rightarrow$ task j starts at time t).

f_{jt}: Binary variable that takes value 1 if task j finishes at the end of period t, and 0 otherwise (that means, $\mathrm{f}_{\mathrm{jt}} = 1 \Rightarrow$ task j finishes at time $t+1$).

T_j: Tardiness of task j.

z_{rt}: Additional capacity of resource r at period t.

3.2.3 Model Formulation

3.2.3.1 Model (1): Single Mode (SM), Single Resource (SR) with No Extra Capacity

Model (1) presents the basic formulation that using a single number of modes *mj*, and a single number of resources *R*, without adding any additional capacity. The Model (1) can be formulated as follows:

$$\text{Model(1)}\ (SM, SR, No\ extra\ capacity)\ : Min \sum_{j=1}^{n} w_j\, T_j \tag{1}$$

Subject to:

$$\sum_{t=1}^{H} S_{jt} = 1, \quad j = 1, \dots, n \tag{2}$$

$$\sum_{t=1}^{H} f_{jt} = 1, \qquad j = 1, \dots, n \tag{3}$$

$$\sum_{t=1}^{H} t\, S_{jt} \geq\ r_j, \qquad j = 1, \dots, n \tag{4}$$

$$\sum_{t=1}^{H} t\, S_{jt} +\ p_j = \sum_{t=1}^{H} t f_{jt}\,, \qquad j = 1, \dots, n \tag{5}$$

$$\sum_{t=1}^{t} S_{jt} - \sum_{t=1}^{t} f_{jt} =\ y_{jt}, \qquad j = 1, \dots, n; t = 1, \dots, H \tag{6}$$

$$\sum_{j=1}^{n} a_j\, y_{jt} \leq b_t, \qquad t = 1, \dots, H \tag{7}$$

$$T_j \geq \sum_{t=1}^{H} t f_{jt} - d_j, \qquad j = 1, \dots, n \tag{8}$$

$$y\ binary, \tag{9}$$

$$T, y, s, f\ \geq 0, \tag{10}$$

The objective function (1) minimizes the total penalty. Constraint (2) requires that each task is assigned exactly to one start time. Constraint (3) requires that each task is assigned exactly to one finish time. Constraint (4) requires that each task is starting at least from the real start date. Constraint (5) enforces that the finish time of a task is equal to the sum of its start time and processing time. Constraint (6) requires that if task j has started processing at time given $\sum_{t=1}^{t} S_{jt} = 1$ and its finishing time at time given $\sum_{t=1}^{t} f_{jt} = 1$, then j is processed during the specified period and y_{jt} = 1. Constraint (7) enforce the resource capacity constraint. Constraint (8) enforce the tardiness constraint. Constraints (9), (10) are both for non-negativity. Table 2 below shows example of SMSR without any additional capacity.

Table 2. Example of SMSR – No Extra Capacity

Project ID	Duration (Months) (p_j)	Capacity (b_t)	Deadline (d_j)	Consumption (a_j)	Weight (w_j)
1	3	7	3	3	1
2	2	4	2	1	1
3	2	6	3	3	1
4	2	6	2	2	1
5	1	8	6	2	1
6	3	8	8	7	1
7	2	10	10	2	1
8	2	8	10	3	1
9	4	8	8	1	1
10	1	5	7	2	1

The exact solution is illustrated in the figure below.

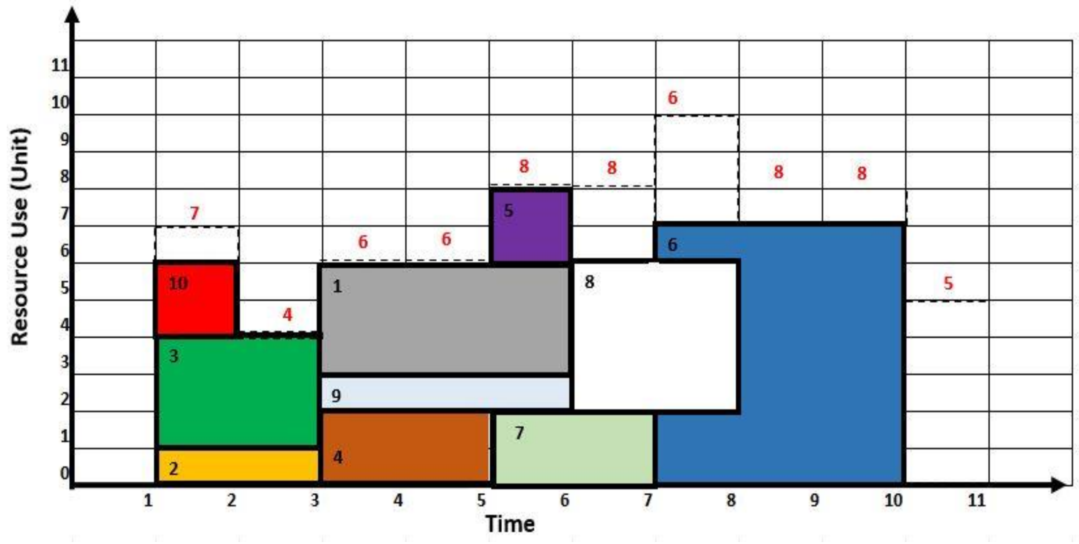

Figure 1. Graphical presentation of SMSR - No Extra capacity

3.2.3.2 Model (2): Single Mode (SM), Single Resource (SR), with Extra Capacity

A variant formulation of Model (1) by adding a new objective function that minimizes the total penalty and any additional capacity, and add constraint (18) which is the capacity constraint can be formulated as follows:

$$\text{Model}(2)\ (SM, SR, FC)\ : Min \sum_{j=1}^{n} w_j\, T_j + \sum_{r=1}^{R} \sum_{t=1}^{H} \sigma_{rt}\, z_{rt} \tag{12}$$

Subject to:

$$\sum_{t=1}^{H} S_{jt} = 1, j = 1, \dots, n \tag{13}$$

$$\sum_{t=1}^{H} f_{jt} = 1, j = 1, \dots, n \tag{14}$$

$$\sum_{t=1}^{H} t\, S_{jt} \geq r_j, \qquad j = 1, \dots, n \tag{15}$$

$$\sum_{t=1}^{H} t\, S_{jt} + p_j = \sum_{t=1}^{H} t f_{jt}, j = 1, \dots, n \tag{16}$$

$$\sum_{t=1}^{t} S_{jt} - \sum_{t=1}^{t} f_{jt} = y_{jt}, \qquad j = 1, \dots, n; t = 1, \dots, H \tag{17}$$

$$\sum_{j=1}^{n} a_j\, y_{jt} \leq b_t + z_{rt}, \qquad t = 1, \dots, H \tag{18}$$

$$T_j \geq \sum_{t=1}^{H} t f_{jt} - d_j, j = 1, \dots, n \tag{19}$$

$$y, z\ binary, \qquad 20$$

$$T, s, f \geq 0, \tag{21}$$

The objective function (12) minimizes the total penalty and any additional capacity. Constraint (13) requires that each task is assigned exactly to one start time. Constraint (14) requires that each task is assigned exactly to one finish time. Constraint (15) requires that each task is starting at least from the real start date. Constraint (16) enforces that the finish time of a task is equal to the sum of its start time and processing time. Constraint (17) requires that if task *j* has started processing at time given $\sum_{t=1}^{t} S_{jt} = 1$ and its finishing time at time given $\sum_{t=1}^{t} f_{jt} = 1$, then j is processed during the specified period and y_{jt} = 1. Constraint (18) enforce the capacity constraint. Constraint (19) enforce the tardiness constraint. Constraints (20), (21) are both for non-negativity.

The number of tasks is 10, The number of resources is 1, Number of execution modes is 1. In this example, we modified the previous example by reducing the capacities of the resource for project 6 and project 7. Table 3 shows example of SMSR with flexible capacity.

Table 3. Example of SMSR – with Extra Capacity

Project ID	Duration (Months) (p_j)	Capacity (b_t)	Deadline (d_j)	Consumption (a_j)	Weight (w_j)
1	3	7	3	3	1
2	2	4	2	1	1
3	2	6	3	3	1
4	2	6	2	2	1
5	1	8	6	2	1
6	3	4	8	7	1
7	2	2	10	2	1
8	2	8	10	3	1
9	4	8	8	1	1
10	1	5	7	2	1

The exact solution is depicted in the figure below.

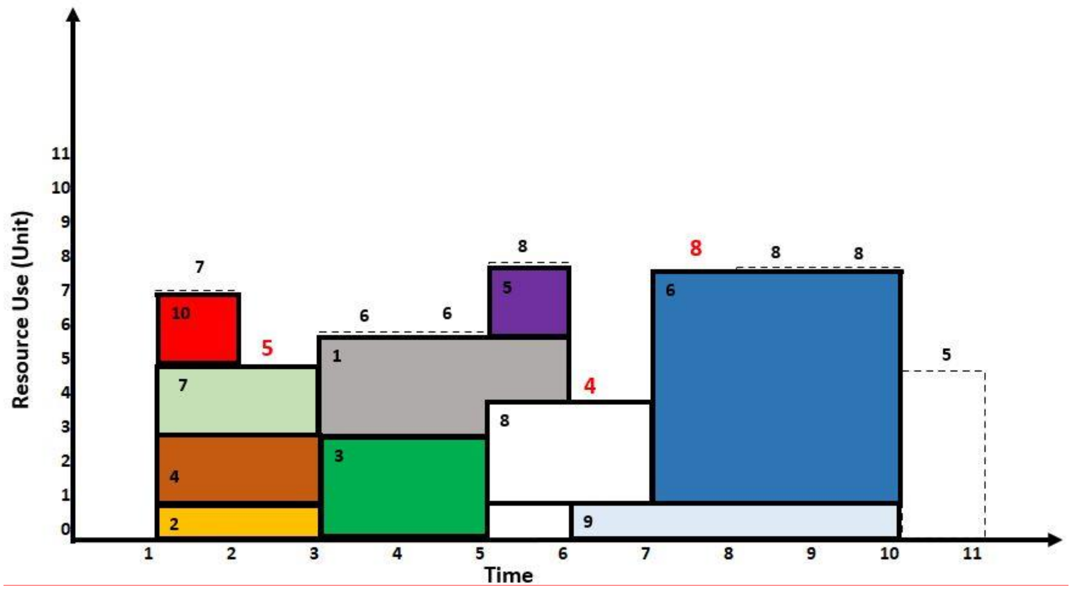

Figure 2. Graphical presentation of SMSR – With Extra capacity

3.2.3.3 Model (3): Single Mode (SM), Multiple Resource (MR), with Extra Capacity

The previous Model (1) and (2) were based on a single resource. We modified the model to solve more complex problems with multiple-number of resources *R*. Model (3) can be formulated as follows:

$$\text{Model(3) } (SM, MR, FC) \ : Min \sum_{j=1}^{n} w_j \, T_j + \sum_{r=1}^{R} \sum_{t=1}^{H} \sigma_{rt} \, z_{rt} \tag{23}$$

Subject to:

$$\sum_{t=1}^{H} S_{jt} = 1, \qquad j = 1, \dots, n \tag{24}$$

$$\sum_{t=1}^{H} f_{jt} = 1, \qquad j = 1, \dots, n \tag{25}$$

$$\sum_{t=1}^{H} t \, S_{jt} \geq \ r_j, \qquad j = 1, \dots, n \tag{26}$$

$$\sum_{t=1}^{H} t \, S_{jt} + \sum_{k=1}^{m_j} \ p_{jk} \ x_{jk} = \sum_{t=1}^{H} t f_{jt} \, , \qquad j = 1, \dots, n \tag{27}$$

$$\sum_{t=1}^{t} \ S_{jt} - \sum_{t=1}^{t} f_{jt} = \ y_{jt}, \qquad j = 1, \dots, n; t = 1, \dots, H \tag{28}$$

$$\sum_{j=1}^{n} \sum_{k=1}^{m_j} a_{jrk} \, x_{jk} \, y_{jt} \leq \ b_{rt} + z_{rt}, \qquad r = 1, \dots, R, ; t = 1, \dots, H \tag{29}$$

$$T_j \geq \sum_{t=1}^{H} t f_{jt} - d_j, \qquad j = 1, \dots, n \tag{30}$$

$$y, z \ \ binary, \tag{31}$$

$$T, s, f \ \geq 0, \tag{32}$$

The objective function (23) minimizes the total penalty and any additional capacity. Constraint (24) requires that each task is assigned exactly to one start time. Constraint

(25) requires that each task is assigned exactly to one finish time. Constraint (26) requires that each task is starting at least from the real start date. Constraint (27) enforces that the finish time of a task is equal to the sum of its start time and processing time. Constraint (28) requires that if task j has started processing at time given $\sum_{t=1}^{t} S_{jt} = 1$ and its finishing time at time given $\sum_{t=1}^{t} f_{jt} = 1$, then j is processed during the specified period and y_{jt} = 1. Constraint (29) enforce the capacity constraint. It is clear that this constraint is not linear. It can be linearized by setting $u_{jkt} = x_{jk}\, y_{jt}$ for j = 1, ..., n; r = 1, ...,R; t = 1, ...,H, and substituting (29) with

$$\sum_{j=1}^{n}\sum_{k=1}^{m_j} a_{jrk}\ u_{jkt} \le b_{rt} + z_{rt} \qquad r = 1, \dots, R, ; t = 1, \dots, H \tag{33}$$

$$x_{jk} + y_{jt} \le u_{jkt} + 1, \qquad j = 1, \dots, n; R = 1, \dots, m_j; t = 1, \dots, H \tag{34}$$

$$Where\ u \ge 0 \tag{35}$$

Constraint (30) enforces for each task the relationship between its tardiness and its corresponding finish time. Constraint (31) enforce linearization. Constraints (31), (32) are both for non-negativity. Constraint (34) requires that if both x_{jk} and y_{jt} take value 1 then u_{jkt} takes value 1 as well.

The number of tasks is 7, the number of resources is 2, Number of execution modes is 1. The capacity b_{rt} for resource #1 is 10, and for resource #2 is 12. Table 4, Shows example of SMMR.

Table 4. Example of SMMR

Project ID	Duration (Months) (p_j)	Start date (r_j)	Deadline (d_j)	R1 (a_{jr})	R2 (a_{jr})	Weight (w_j)
1	15	15	35	3	3	3
2	6	10	17	3	4	4
3	5	1	6	1	4	4
4	6	7	16	4	5	5
5	9	12	22	5	3	4
6	13	18	31	4	1	1
7	16	13	29	2	5	3

The figure below shows the optimal allocation of resource #1.

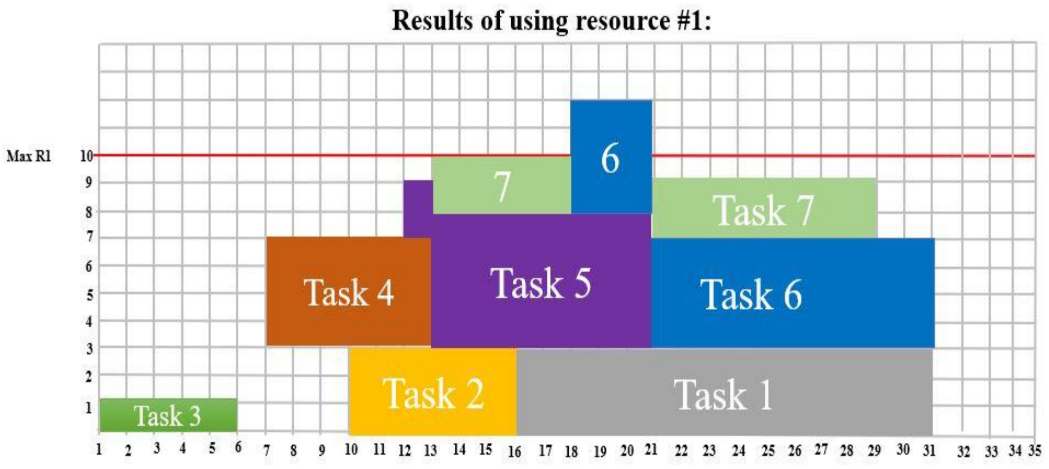

Figure 3. Graphical presentation of SMMR for resource #1

The figure below shows the optimal allocation of resource #2.

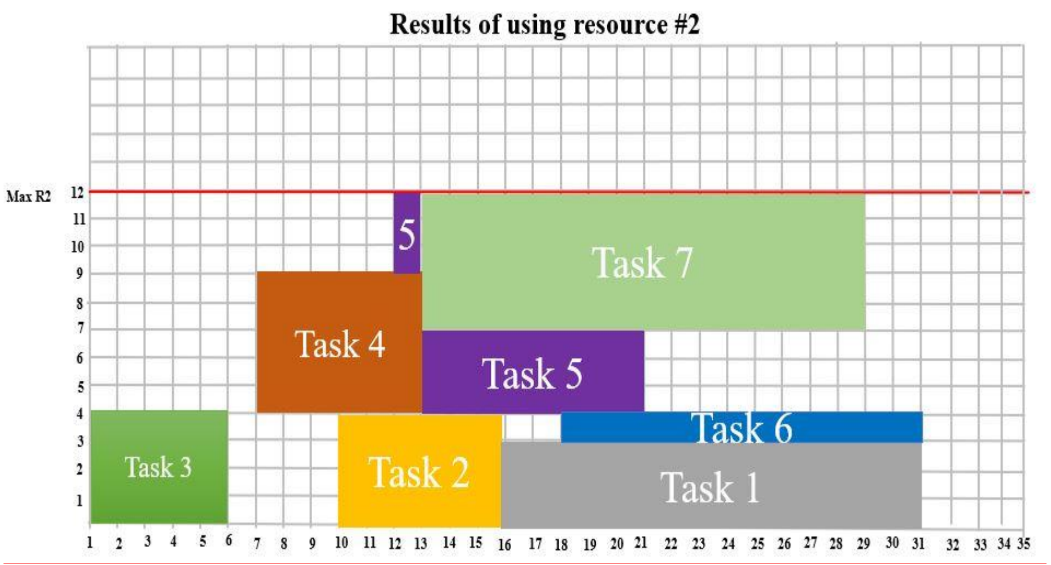

Figure 4. Graphical presentation of SMMR for resource #2

3.2.3.4 Model (4): Multiple modes (MM), Multiple Resources (MR), with Extra Capacity

Model (4) is the last model that aims to solve problems with multiple execution modes m_j and multiple numbers of resources *R*. It can be formulated as follows:

$$\text{Model}(4)\ (MM, MR, FC)\ : Min \sum_{j=1}^{n} w_j\, T_j + \sum_{r=1}^{R}\sum_{t=1}^{H} \sigma_{rt}\, z_{rt} \qquad (36)$$

Subject to:

$$\sum_{k=1}^{m_j} X_{jk} = 1, \qquad j = 1, \dots, n \qquad (37)$$

$$\sum_{t=1}^{H} S_{jt} = 1, \qquad j = 1, \dots, n \qquad (38)$$

$$\sum_{t=1}^{H} f_{jt} = 1, \qquad j = 1, \dots, n \qquad (39)$$

$$\sum_{t=1}^{H} t\, S_{jt} \geq r_j, \qquad j = 1, \dots, n \qquad (40)$$

$$\sum_{t=1}^{H} t\, S_{jt} + \sum_{k=1}^{m_j} p_{jk}\ x_{jk} = \sum_{t=1}^{H} tf_{jt}\,, \qquad j = 1, \dots, n \tag{41}$$

$$\sum_{t=1}^{t} S_{jt} - \sum_{t=1}^{t} f_{jt} = y_{jt}, \qquad j = 1, \dots, n; t = 1, \dots, H \tag{42}$$

$$\sum_{j=1}^{n} \sum_{k=1}^{m_j} a_{jrk}\ u_{jkt} \le b_{rt} + z_{rt} \quad r = 1, \dots, R, ; t = 1, \dots, H \tag{43}$$

$$T_j \ge \sum_{t=1}^{H} tf_{jt} - d_j, \qquad j = 1, \dots, n \tag{44}$$

$$x_{jk} + y_{jt} \le u_{jkt} + 1, \qquad j = 1, \dots, n; R = 1, \dots, m_j; t = 1, \dots, H \tag{45}$$

$$x, y, z\ \ binary, \tag{46}$$

$$T, s, f, u \ge 0, \tag{47}$$

The objective function (34) minimizes the total penalty and any additional capacity. Constraint (35) requires that each task is assigned exactly to one mode. Constraint (36) requires that each task is assigned exactly to one start time. Constraint (37) requires that each task is assigned exactly to one finish time. Constraint (38) requires that each task is starting at least from the real start date. Constraint (39) enforces that the finish time of a task is equal to the sum of its start time and processing time. Constraint (40) requires that if task *j* has started processing at time given $\sum_{t=1}^{t} S_{jt} = 1$ and its finishing time at time given $\sum_{t=1}^{t} f_{jt} = 1$, then j is processed during the specified period and y_{jt} = 1. Constraint (41) enforce the capacity constraint. Constraint (42) enforce the tardiness constraint. Constraint (43) enforce linearization. Constraints (44), (45) are both for non-negativity.

The number of tasks is 7, the number of resources is 2, Number of execution modes is 2. The capacity b_{rt} for resource #1 is 10, and for resource #2 is 12. Table 5 Shows example of MMMR.

Table 5. Example of MMMR

Project ID	Mode 1 Duration (Months) (p_j)	Mode 2 Duration (Months) (p_j)	Start date (r_j)	Deadline (d_j)	R1	R2	R1	R2	Weight *(w_j)*
1	15	20	15	35	3	3	1	3	3
2	6	7	10	17	3	4	3	4	4
3	5	4	1	6	1	4	1	4	4
4	6	9	7	16	4	5	4	5	5
5	9	10	12	22	5	3	2	3	4
6	13	12	18	31	4	1	2	1	1
7	16	15	13	29	2	5	2	2	3

The optimal resource 1 allocation is shown in the figure below.

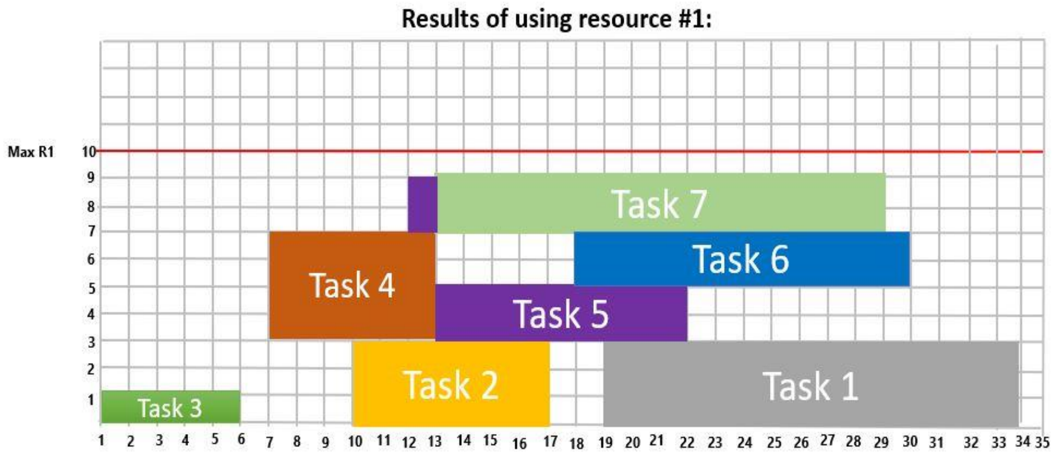

Figure 5. Graphical presentation of MMMR for resource #1

The optimal resource 2 allocation is depicted in the figure below.

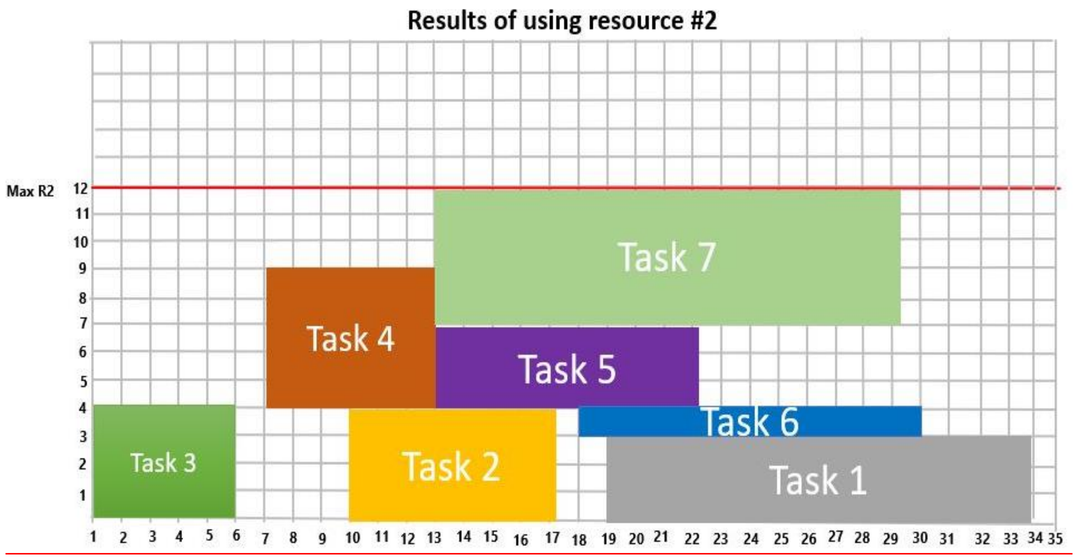

Figure 6. Graphical presentation of MMMR for resource #2

3.2.3.5 Conclusion

This chapter proposed a mathematical model for the Multiple Resource Loading Problem (MRLP). The notation was described, and the decision variables were defined. The model formulation process was outlined which proved to be a challenging undertaking. The MRLP model formulation was undertaken in four models: SMSR with no extra capacity, SMSR with flexible capacity, SMMR model with flexible capacity, and finally MMMR model with flexible capacity. For all four models, different levels of constraints were used: time constraint, capacity constraint, tardiness. Constraints were also used to enforce linearization and non-negativity. The following chapter (4) present computational experiments to show the efficacy of the proposed formulations.

CHAPTER 4: COMPUTATIONAL EXPERIMENTS

4.1 Introduction

This chapter gives a detailed overview of the computational experiments that were performed in the proposed model. The objective of the computational experiments was to investigate the effectiveness of the models. In particular, the proposed MRLP was coded and implemented using real and randomly generated test instances. The following sections explain the test instance implementation.

4.2 Implementation

To evaluate the model's empirical efficiency, the proposed mixed-integer problem was implemented using computer software. The problem was coded with Eclipse IDE for Java Developers 2019-12 (4.14.0) Version and was solved by using IBM ILOG CPLEX Optimization Studio 20.1.0.0 version. The coding was done on Windows 10 operating system with Intel i7@1.80 GHz, and 8.00 GB of RAM.

4.2.1 Optimization Programming Language (OPL)

The computational experiments also make use of optimization programming language (OPL). OPL is a modeling tool for optimization problems that make use of algebraic primitives and facilitates direct mapping of decision variables, sets, constraints, and parameters. The IDE (Integrated Development Environment) is available under OPL. Moreover, OPL is included in the CPLEX Studio package. Within the BM Decision Optimization product family, users can either choose OPL or other programming languages such as C+, Java, and Python.

4.2.2 CPLEX

IBM ILOG CPLEX Optimization Studio (CPLEX) is widely used to solve complex optimization problems. The CPLEX program relies on constraint programming techniques to solve mixed-integer optimization problems. CPLEX also contains the

ILOG CP Optimizer which is used to solve combinatorial optimization problems that cannot be easily linearized with normal optimization programs.

4.2.3 Description of the test instances

The model was tested for effectiveness and efficiency. The current model used data provided by the Ministry of Administrative Development, Labor and Social Affairs (ADLSA) in Qatar. Ministry officials were approached and were requested to provide the data regarding multiple resource planning in the ministry. Real-life data was provided. The data from the ministry was used to randomly generate test instances. All instances generated incorporated factors such as number of available resources, number of modes, number of tasks, processing time, starting date, deadline required for the completion of specific tasks, the weight of each task, and consumption needed from each resource. The number of projects ranges from 10, 20, 30, 50, 100, and 200 projects. For each project size, there are 10 test instances generated randomly by using the uniform distribution. The processing time in weeks is between 6 and 104 weeks. The duration in months can be obtained by dividing the duration in weeks by 4. The starting date is between 1 and 24. The deadline is calculated by summation the duration in months with the real start date. Resource #1 consumptions are between 1 and 4. On the other hand, resource #2 consumptions are between 1 and 6. Finally, the priority for all projects is between 1 and 5.

4.2.4 Performance of the proposed model

The results of our proposed model were assessed by the output of the codes implemented in the IBM ILOG CPLEX Optimization Studio 20.1.0.0 version. Coding was performed on Windows 10 operating system with Intel i7@1.80 GHz, and 8.00 GB of RAM. The results of the model are computed in 6 different sizes of instances ranging from 10 projects up to 200 projects.

4.2.4.1 Solving Group 1 Test Instances

Group 1 is the smallest problem size based on the number of tasks which is 10 tasks. The number of resources for this group is 2. Solving this group was by using the real data provided by (ADLSA) to randomly generate 10 test instances, each instance has different values of parameters such as processing time (months), start date, due date, consumption needed from resource #1, and resource #2, weight, and calculating constraints for resource capacity each project. In reasonable time, all instances have been solved to optimality. Table 6 and 7 shows results of group 1 test instances.

Table 6. Results of Group 1 Test Instances (10 Projects)

Instance	# of constraints	# of variables	Run time (S)
1	482	1162	1.32
2	554	1354	1.28
3	554	1354	1.17
4	482	1162	1.05
5	458	1098	1
6	602	1482	1.11
7	542	1322	0.95
8	518	1258	0.99
9	590	1450	0.94
10	518	1258	0.92

Table 7. Group 1 Test Instances CPU Time (10 Projects)

CPU	Run time (S)
Avg.	1.073
Max.	1.32
Min.	0.92

4.2.4.2 Solving Group 2 Test Instances

In Group 2, the number of tasks was increased to 20 projects. The number of resources was set to 2. 20 instances were generated randomly with different parameter values. In reasonable time, all instances have been solved to optimality. It is important to mention that the maximum run time increased from 1.32 to 2.28 seconds. Table 8 and 9 displays the results.

Table 8. Results of Group 2 Test Instances (20 Projects)

Instance	# of constraints	# of variables	Run time (S)
1	1178	3058	2.28
2	1024	2624	1.44
3	1134	2934	1.67
4	1134	2934	1.72
5	1178	3058	1.58
6	980	2500	1.53
7	1112	2872	1.72
8	892	2252	1.45
9	1024	2624	1.46
10	1090	2810	1.56

Table 9. Group 2 Test Instances CPU Time (20 Projects)

CPU	Run time (S)
Avg.	1.64
Max.	2.28
Min.	1.44

4.2.4.3 Solving Group 3 Test Instances

Moving to Group 3, the number of tasks was increased to 30 while keeping the same number of resources. Maximum running time very similar to the previous group. All instances were decided to optimize in a reasonable time. The results are shown in Table 10 and 11.

Table 10. Results of Group 3 Test Instances (30 Projects)

Instance	# of constraints	# of variables	Run time (S)
1	1526	3986	2.01
2	1750	4630	2.23
3	1590	4170	1.91
4	1590	4170	2.08
5	1558	4078	2.23
6	1398	3618	2.02
7	1654	4354	2.14
8	1750	4630	2.19
9	1430	3710	1.82
10	1430	3710	1.96

Table 11. Group 3 Test Instances CPU Time (30 Projects)

CPU	Run time (S)
Avg.	2.056
Max.	2.23
Min.	1.82

4.2.4.4 Solving Group 4 Test Instances

As for Group 4, the number of tasks increased to become 50 tasks in total, while maintaining the resource number to 2. By adding more tasks, the maximum running time slightly increased than the previous group. All instances have been solved to optimality. The results are shown in Table 12 and 13.

Table 12. Results of Group 4 Test Instances (50 Projects)

Instance	# of constraints	# of variables	Run time (S)
1	2642	7042	2.39
2	2694	7194	2.66
3	2746	7346	2.92
4	2694	7194	2.66
5	2746	7346	3.2
6	2746	7346	2.76
7	2642	7042	2.32
8	2746	7346	2.5
9	2694	7194	2.5
10	2694	7194	2.36

Table 13. Group 4 Test Instances CPU Time (50 Projects)

CPU	Run time (S)
Avg.	2.627
Max.	3.2
Min.	2.32

4.2.4.5 Solving Group 5 Test Instances

In group 5, the number of tasks set to 100 tasks in total. Running time in this case has a slight rise than the previous group of test instances. All instances have been solved to optimality. The results are shown in Table 14 and 15.

Table 14. Results of Group 5 Test Instances (100 Projects)

Instance	# of constraints	# of variables	Run time (S)
1	5498	14898	3.31
2	5294	14294	3.45
3	5294	14294	3.31
4	5294	14294	3.19
5	5498	14898	3.89
6	5192	13992	3.68
7	5294	14294	3.38
8	5090	13690	3.15
9	5396	14596	3.09
10	5396	14596	3.44

Table 15. Group 5 Test Instances CPU Time (100 Projects)

CPU	Run time (S)
Avg.	3.389
Max.	3.89
Min.	3.09

4.2.4.6 Solving Group 6 Test Instances

As for group 6, the number of tasks was increased to solve extremely large-scale problems, we set the number of tasks to be 200 tasks in total. Adding has increased the running time. In this case, we found that the maximum time was 5.53 seconds. All instances have been solved to optimality. The results are shown in Table 16 and 17.

Table 16. Results of Group 6 Test Instances (200 Projects)

Instance	# of constraints	# of variables	Run time (S)
1	10898	29698	3.63
2	10898	29698	4.79
3	10292	27892	4.73
4	10494	28494	4.72
5	10898	29698	5.36
6	10696	29096	5.07
7	10696	29096	4.78
8	10494	28494	4.95
9	10898	29698	5.05
10	10292	27892	5.53

Table 17. Group 6 Test Instances CPU Time (200 Projects)

CPU	Run time (S)
Avg.	4.861
Max.	5.53
Min.	3.63

4.2.4.7 Results Summary

As illustrated in Table 20, the average results from running the model instances were solved in between 1.073 and 4.861 seconds. It is worth mentioning that the maximum running time was 5.53 seconds for group 6 with up to 200 projects and on average it takes less than 4.9 seconds to solve the largest problem. Table 18 shows summary of CPU running time of all instances.

Table 18. Summary of CPU Running Time of All Instances

Problem size		CPU time	
	Avg.	Max.	Min.
10	1.073	1.32	0.92
20	1.641	2.28	1.44
30	2.056	2.23	1.82
50	2.627	3.2	2.32
100	3.389	3.89	3.09
200	4.861	5.53	3.63

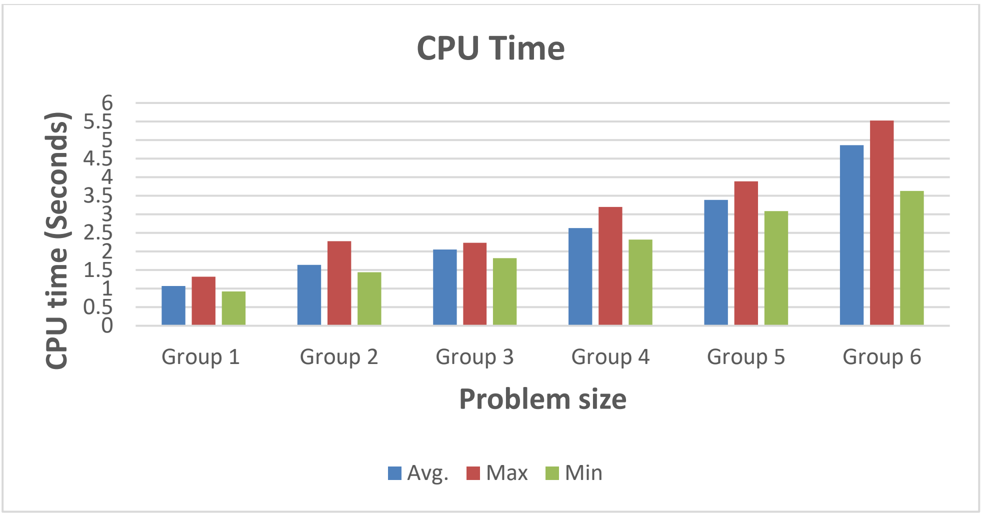

Figure 7. CPU time for all groups test instances

To conclude, Figure 7, shows a summary of the main results of CPU time for the proposed model of all computational experiments groups. This chart figures out the average, maximum, and minimum run time for each group. The run time difference between all experiments for the six groups. After testing all instances, and as the complexity of the test instances increased, the CPU time has increased reasonably. The proposed model was performed within a very short time (seconds) even when increasing complexity of tasks number to 200, the maximum running time was completed in 5.53 seconds while the average time-solving time is between 1.073 and 4.861 seconds.

4.2.4.8 Solving Multi-Mode, Multi-Resource (MMMR) Model with Extra Capacity

In this model, we solved the multi-mode multi-resource problem where we added more complexity to the model by increasing the number of execution modes to become multiple modes. We implemented the model by testing 2 groups of test instances, first group includes 7 tasks the second group has 20 tasks. Each group has different parameter values. In reasonable time, all instances have been solved to optimality. Table 18. shows results of multi-mode, multi-resources model.

Table 19. Results of Multi-Mode Multi-Resources Model

Problem size	# of constraints	# of variables	Run time (S)
7 (instance 1)	847	1316	1.29
7 (instance 2)	847	1316	1.48
20	2910	4650	9.11

CHAPTER 5: CONCLUSION AND FUTURE RESEARCH

In this paper, we presented the optimization models for multiple resource loading (MRLP) to gives an overview and provides a solution that minimizes time wastages and ensures efficient resource utilization for both single-mode and multi modes of projects, while allows the combination of resources while considering restrictions and time constraints to achieve optimal scheduling and resource planning. Our proposed model applying the different enhancement procedures to fill the gap in the literature and provide an efficient solution for multiple-mode with multiple resources large-scale instances.

Real-life data was provided by the Ministry of Administrative Development (ADLSA) to randomly generate test instances for (MRLP) models. Coding was performed on Windows 10 by using IBM ILOG CPLEX Optimization Studio 20.1.0.0 version. operating system with Intel i7@1.80 GHz, and 8.00 GB of RAM. We generated six different groups of instances for the single-mode, multiple resource models while using flexible extra capacity (SM, MR, FC). Then, we generated two groups to verify the multiple-mode, multiple-resource model with flexible extra capacity (MMMR). All instances generated different parameters such as number of available resources, number of execution modes, number of tasks, processing time, starting date, deadline required for the completion of specific tasks, the weight of each task, and consumption needed from each resource. Furthermore, we illustrated the main results of CPU time for the proposed model of all computational experiments groups and shows the effectiveness of the proposed models.

Moving to the sustainable aspect, optimization models for multiple resource planning aims to minimize time wastages and ensures efficient resource planning. By applying these kinds of models, it ensures the best allocation of renewable resources such as employees and machines, that minimize the impact of both energy and economic aspects that may affect the total cost of completing projects and avoid any tardiness that may cost some penalties.

Finally, as for future research directions, our paper provides a strong ground for future research the model can be modified to adding further complexity to the problem such as solving large scale instance, number of projects, number of execution modes, number of resources and distributed tasks in a specific time horizon to ensures efficient of resource utilization.

REFERENCES

1. Beliën, J., Cardoen, B., & Demeulemeester, E. (2012). Improving workforce scheduling of aircraft line maintenance at Sabena Technics. *Interfaces*, *42*(4), 352-364.
2. Bianco, L., & Caramia, M. (2013). A new formulation for the project scheduling problem under limited resources. *Flexible Services and Manufacturing Journal*, *25*(1-2), 6-24.
3. Blazewicz, J., Machowiak, M., Weglarz, J., Kovalyov, M. Y., & Trystram, D. (2004). Scheduling malleable tasks on parallel processors to minimize the makespan. *Annals of Operations Research*, *129*(1-4), 65.
4. Brucker, P. and Knust, S. (2012). Complex Scheduling. Springer
5. Boctor F.F, “An adaption of the simulated annealing algorithm for solving the resource-constrained project scheduling problems,” *Int. J. Prod. Res.*, vol. 34, pp. 2335–2351, 1996.
6. Burgelman, J., & Vanhoucke, M. (2018). Maximising the weighted number of activity execution modes in project planning. *European Journal of Operational Research*, *270*(3), 999-1013.
7. Carruthers, J. A., and Battersby, A. (1966). “Advances in Critical Path Methods.” Operational Research Quarterly, 17(4): 359-380.
8. Chakrabortty, R., Abbasi, A., & Ryan, M. (2019). Multi-mode resource-constrained project scheduling using modified variable neighborhood search heuristic. *International Transactions in Operational Research*, *27*(1), 138-167. doi: 10.1111/itor.12644.

9. Christofides, N., Alvarez-Valdes, R., and Tamarit, J. M. (1987). “Project Scheduling with Resource Constraints: A Branch and Bound Approach.” European Journal of Operational Research, 29: 262-273.
10. Cooper, D. F. (1974). “An Experimental Investigation of Some Heuristics for Scheduling Resource-constrained Project.” Ph.D. Thesis, Department of Computer Science, University of Adelaide.
11. Davis, E. W., and Heidorn, G. E. (1971). “An Algorithm for Optimal Project Scheduling under Multiple Resource Constraints.” Management Science, 17(12): B803-B816.
12. De Nijs, F. and Klos, T. (2014). A novel priority rule heuristic: Learning from justification. In Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling (ICAPS’14), pages 92–100.
13. De Boer, R. (1998). *Resource-constrained multi-project management* (Doctoral dissertation, Ph.D. thesis, University of Twente, The Netherlands).
14. Dorndorf, U., Pesch, E., and Phan-Huy, T. (2000). “A Branch-and-Bound Algorithm for the Resource-Constrained Project Scheduling Problem.” Mathematical Methods of Operations Research, 52: 413-439.
15. Dorigo, M., Maniezzo, V., and Colorni, A. (1996). “Ant System: Optimization by a Colony of Cooperating Agents.” IEEE Transactions on Systems, Man, and Cybernetics-Part B, 26(1): 29-41.
16. Dooley, G. Lupton, and D. O’Sullivan (2005), “Multiple project management: a modern competitive necessity,” J. Manuf. Technol. Manag., vol. 16, no. 5, pp. 466–482, Jul.
17. Drexl A. and J. Grunewald (1993), “Nonpreemptive multi-mode resource-constrained project scheduling,” IIE Trans., vol. 25, pp. 74–81, 1993.

18. Fisher, M. (1973). “Optimal Solution of Scheduling Problems using Lagrange Multipliers.” Part I: Operations Research, 21: 1114-1127. Part II: in S. E. Elmaghraby (ed.), Symposium on the Theory of Scheduling and its Applications, Springer, Berlin, 294-318.
19. Fündeling, C. U., & Trautmann, N. (2010). A priority-rule method for project scheduling with work-content constraints. *European Journal of Operational Research*, *203*(3), 568-574.
20. Gademann, N., & Schutten, M. (2005). Linear-programming-based heuristics for project capacity planning. *Iie Transactions*, *37*(2), 153-165.
21. Hartmann, S., 2001. Project scheduling with multiple modes: a genetic algorithm. Annals of Operations Research 102, 1–4, 111–135.
22. Harris R.B. (1973), Precedence and Arrow Networking Techniques for Construction. University of Michigan.
23. Hariga and S. M. El-Sayegh, “Cost Optimization Model for the Multiresource Leveling Problem with Allowed Activity Splitting,” J. Constr. Eng. Manag., vol. 137, no. 1, Jan. 2011.
24. Hans, E. (2001). Resource loading by branch-and-price techniques (Ph. D. thesis). *The Netherlands: University of Twente*.
25. Hastings, N. A. J. (1972). “On Resource Allocation in Project Networks.” Operational Research Quarterly, 23(2): 217-221.
26. Holland, J. K. (1975). Adaptation in Neural and Artificial Systems, University of Michigan Press, Ann Arbor, MI.
27. Kis, T. (2005). A branch-and-cut algorithm for scheduling of projects with variable-intensity activities. *Mathematical programming*, *103*(3), 515-539.

28. Liao T.W, P. J. Egbelu, B. R. Sarker, and S. S. Leu, “Metaheuristics for project and construction management – A state-of-the-art review,” Autom. Constr., vol. 20, no. 5, pp. 491–505, Aug. 2011.
29. Merkle, D., Middendorf, M., and Schmeck, H. (2002). “Ant Colony Optimization for Resource-Constrained Project Scheduling.” IEEE Transactions on Evolutionary Computation, 6(4): 333-346.
30. Mingozzi, A., Maniezzo, V., Ricciardelli, S., and Bianco, L. (1998). “An Exact Algorithm for the Resource-Constrained Project Scheduling Problem Based on a New Mathematical Formulation.” Management Science, 44(5): 714-729.
31. Mohammed A. Salem Hiyassat (2000), “Modification of Minimum Moment Approach in Resource Leveling,” J. Constr. Eng. Manag., vol. 126, no. 4, pp. 278–284, Jul.
32. Naber, A., & Kolisch, R. (2014). MIP models for resource-constrained project scheduling with flexible resource profiles. *European Journal of Operational Research*, *239*(2), 335-348.
33. Nattaf, M., Horváth, M., Kis, T., Artigues, C., & Lopez, P. (2019). Polyhedral results and valid inequalities for the Continuous Energy-Constrained Scheduling Problem. *Discrete Applied Mathematics*, *258*, 188-203.
34. Patterson, J. H., and Huber, W. D. (1974). “A Horizon-Varying, Zero-One Approach to Project Scheduling.” Management Science, 20(6): 990-998.
35. Patterson, J. H., and Roth, G. W. (1976). “Scheduling a Project under Multiple Resource Constraints: A Zero-One Programming Approach.” AIIE Transactions, 8: 449-455.

36. Palpant, M., Artigues, C., and Michelon, P. (2004). “LSSPER: Solving the Resource-Constrained Project Scheduling Problem with Large Neighbourhood Search.” Annals of Operations Research, 131: 237-257.
37. Project Management Institute. A Guide to the Project Management Body of Knowledge, Fifth Edition. Newtown Square, Pa.: Project management Institute, ©2013, 2013.
38. Rieck. J and Zimmermann J. (2015), “Exact Methods for Resource Leveling Problems,” in Handbook on Project Management and Scheduling Vol.1, C. Schwindt and J. Zimmermann, Eds. Cham: Springer International Publishing, pp. 361–387.
39. Ramlogan. R and I. C. Goulter, “Mixed Integer Model for Resource Allocation in Project Management,” Eng. Optim., vol. 15, no. 2, pp. 97–111, Dec. 1989.
40. Schutt, A., Feydy, T., Stuckey, P. J., and Wallace, M. G. (2013). Solving rcpsp/max by lazy clause generation. Journal of Scheduling, 16(3):273–289.
41. Senouci A.B and N. N. Eldin (2004), “Use of Genetic Algorithms in Resource Scheduling of Construction Projects | Journal of Construction Engineering and Management | Vol 130, No 6,” J. Constr. Eng. Manag., vol. 130, no. 6.
42. Shirzadeh Chaleshtari, A. (2017). Resource tardiness weighted cost minimization in project scheduling. *Advances in Operations Research*, *2017*.
43. Song, G., Kis, T., & Leus, R. (2019). Polyhedral results and branch-and-cut for the resource loading problem. *FEB Research Report KBI_1908*.
44. Sousa, J. P., & Wolsey, L. A. (1992). A time indexed formulation of non-preemptive single machine scheduling problems. *Mathematical programming*, *54*(1-3), 353-367.

45. Ulusoy, G., and Özdamar, L. (1989). “Heuristic Performance and Network/Resource Characteristics in Resource-Constrained Project Scheduling.” The Journal of the Operational Research Society, 40(12): 1145-1152.

46. Wellman, M. P., Walsh, W. E., Wurman, P. R., and MacKie-Mason, J. K. (2001). Auction protocols for decentralized scheduling. Games and economic behavior, 35(1):271–303.

47. Wullink, G., Gademann, A. J. R. M., Hans, E. W., & van Harten, A. (2004). Scenario-based approach for flexible resource loading under uncertainty. *International Journal of Production Research, 42*(24), 5079-5098.

48. Varakantham, P., Fu, N., and Lau, H. C. (2016). A proactive sampling approach to project scheduling under uncertainty. In Proceedings of the Thirtieth national conference on Artificial intelligence (AAAI’16), pages 3195–3201.