AN ABSTRACT OF THE DISSERTATION OF

Yasaman Mehravaran for the degree of Doctor of Philosophy in
Industrial Engineering presented on April 17, 2013.
Title: Hybrid Flowshop Scheduling with Dual Resources in a Supply Chain

Abstract approved: ____
Rasaratnam Logendran

This dissertation addresses a hybrid-flow shop scheduling problem with dual resource constraints in a supply chain. Most of the traditional scheduling problems deal with machine as the only resource. However, other resources such as labor is not only required for processing jobs but are often constrained. Considering the second resource (labor) makes the scheduling problems more realistic and practical to implement in industries. In this research labor has different skill levels and the skill level required to perform the setup could be different from that needed to perform the run. The setup time is sequence-dependent, and job release times and machine availability times are dynamic. Also machine skipping is allowed.

In tactical supply chain decisions such as scheduling, the goal is to minimize the cost of producer. However, when looking at the whole network, minimizing the cost of the producer alone may not lead to minimizing the cost of the whole supply chain. In fact the coordination between the producer and other entities in the network can minimize the cost. In this dissertation coordination between producer and customers is considered in order to make effective scheduling decisions. The goal of this research is to minimize the work-in-process inventory for the producer and maximize customers' service level to maintain producer-customers coordination.

A linear mixed-integer mathematical programming model is proposed and CPLEX solver is used to find solutions for generated example problems with branch-and-bound technique. As the problem is NP-hard in the strong sense three different meta-search heuristic algorithms based on tabu search are developed in order to quickly solve the scheduling problems. A total of 243 examples were generated in small, medium and large size problems. Search algorithms performance in small size problems can be assessed by comparing them with the optimal solution from branch-and-bound method. However, in medium and large size problems, branch-and-bound method cannot find the optimal solution and therefore for assessing the performance of search algorithms three different lower bounding methods are proposed. The first method is based on Logic-Based Benders Decomposition and the second and third methods are two different variations of iterative selective linear programming (LP) relaxation called fractional LP relaxation and positive LP relaxation.

An experimental analysis based on a nested-factorial design with blocking is developed in order to identify statistically significant differences between the effectiveness and efficiency of the lower bounding methods and search algorithms. The results showed that the proposed search algorithms and lower bounding methods are very effective and efficient. On average the developed lower bounding methods tighten the lower bound found by branch-and-bound by 11.93%. The quality of search algorithms is the same as the upper bound found by branch-and-bound. However, the search algorithms are on average 3.8 times faster than the branch-and-bound method.

Hybrid Flowshop Scheduling with Dual Resources in a Supply Chain

by

Yasaman Mehravaran

A DISSERTATION

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Presented April 17, 2013
Commencement June 2013

Doctor of Philosophy dissertation of Yasaman Mehravaran presented on April 17, 2013.

APPROVED:

Major Professor, representing Industrial Engineering

Head of the School of Mechanical, Industrial and Manufacturing Engineering

---

Dean of the Graduate School

I understand that my dissertation will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my dissertation to any reader upon request.

Yasaman Mehravaran, Author

ACKNOWLEDGMENTS

I would like to thank my major professor, Dr. Rasaratnam Logendran, for his support and continued guidance over the last four years. He was always there to answer my questions 24/7 even on the weekends and holidays. His persistence, enthusiasm and hard work were truly a motivation for me to continue and not to get disappointed. I want to thank him for his moral support and also for being a good friend.

My gratitude extends to my committee members Dr. Zhaohui Wu, Dr. David Porter, and Dr. Karl Haapala, for their guidance and support. Special thanks to Dr. Wu for serving as my minor professor and for his constant encouragement and useful feedback. I would like to thank Dr. James Coakley Jr. and Dr. Prasad Tadepalli for serving as the graduate council representative on my committee. I would like to express my gratitude to Dr. Wade Marcum for taking over their position on my committee.

I would like to express my appreciation to the MIME staff members Jean Robinson and Phyllis Helvie for their help. Special thanks to Keith Price who installed and maintained any software/hardware that I needed for my research.

A number of people made the last four years bearable with their keen friendship. I would like to thank Hadi, Abbas, and Mohammad for their help and understanding. I thank my close friend Morvarid for her constant support and long phone conversations we had whenever I needed.

I will always be grateful to Jason for his love and support and constantly cheering me up with his presence.

Last, but certainly not least, I would like to thank my mom, my brother and my sister for their everlasting love and care.

TABLE OF CONTENTS

Page

Page

TABLE OF CONTENTS (Continued)

Page

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF TABLES

Table | Page

LIST OF TABLES (Continued)

LIST OF TABLES (Continued)

LIST OF TABLES (Continued)

LIST OF APPENDICES

LIST OF APPENDIX TABLES

DEDICATION

To my mom and the memory of my dad

and to Jason

Hybrid Flowshop Scheduling with Dual Resources in a Supply Chain

1. INTRODUCTION

Sequencing and scheduling has been one of the topics of interest for many researchers in the past few decades. In traditional scheduling problems, the only resource considered as a constraint is machine. However, in reality other resources such as labor and tools are constrained and it is illogical to consider that there is always enough labor available for processing a job. Most of the previous research assumes that the only resource that is constrained in scheduling problems is the machine and there is enough labor available to process the jobs [1], which is clearly not a true statement. Today's industries are concerned about their labor cost as much as they are concerned about their machine cost. Although the cost of machines is higher compared to the cost of labor, the latter can be significant, and thus saving labor cost is in manufacturer's interest. Considering labor as a resource in addition to machine makes the scheduling problem even more complex and it is more complicated to assign resources to jobs. In other words, it is more difficult to assign resources in problems with dual resource constraints than problems with machine as the only constraint. In a dual resource constrained scheduling problem, the machine can be idle either due to the unavailability of jobs or the unavailability of labor. Usually in dual-resource constrained scheduling problems, labor is more constrained than machines. Gargeya and Deane [1] reported that in most of the dual-resource job shops, labor utilization is 80%-90% while machine utilization varied from 45% to 90%. Introducing an additional resource changes the development of the methodology required to investigate the scheduling problem. A paradigm shift is needed in order to calculate the completion time compared to single resource. In dual resource problems, in addition to the schedule of jobs on machines, the schedule of jobs on labor and machine-labor schedule interferences have to be considered. This means that for a specific sequence of jobs on machine, there is an optimal sequence of jobs on labor. Conversely, for a specific sequence of jobs on labor, there is an optimal sequence of jobs on machines. By adding an additional resource to a scheduling problem, the complexity of the problem can be viewed as being more than two fold.

The problem presented here is inspired from a real project at Davis Tool (DT) Company, Portland, Oregon. The project proposed by DT had dual-resource constraints (labor and machine) in which labor is more constrained than machines. Different labor skills are available and the skill required for performing the setup is equal or higher than the skill required for performing the run. Implementing these characteristics to the scheduling problem is of significant importance as the use of labor with a higher skill would cost more than the others. The machines and jobs are also dynamic, which means that not all the jobs and machines are available at the start of the planning horizon. The problem at DT can be described as a two-stage flowshop with unrelated-parallel machines and sequence-independent setup times. In the problem that we are proposing here, all the characteristics of the DT problem such as the dual resource constraints, use of different skills for performing job setup and run, and dynamic machine availability and job release times have been kept. To generalize the proposed research problem and make it applicable to numerous scheduling problems found in the industry, a hybrid-flowshop, comprised of multiple-stages (i.e., two or more), is considered, instead of a two-stage problem as that in DT. Another operational issue considered to generalize the proposed research problem is to assume that the setup times are sequence-dependent. In industry scheduling, the job setup time on a machine typically depends on the previously processed job on that machine. If the jobs are similar, the setup time is short and if the jobs are different, a long setup is needed. When the setup times are close to each other or in other words the variance of setup times is small, the average setup time is used as the sequence-independent setup time. However, this is not the case if the setup variance is high and, in such instances, the companies need to rely on the use of sequence-dependent setup times. In other words, a problem that considers sequence-dependent setup time can easily be adapted and made applicable to a sequence-independent setup time problem, but not the other way around. Clearly, the DT problem can be viewed as a special case of the generalized problem that is being proposed in this research.

1.1. Research Contributions

This dissertation specifically addresses hybrid flowshop scheduling with dual resources and is motivated by a real industry application. The focus is on bicriteria objectives of minimizing the sum of completion time and minimizing the sum of tardiness. Different skill levels are considered for labor and the skill required for performing the setup can be different from the skill required for performing the run. The problem investigated in this dissertation is shown to be NP-hard in the strong sense with the bicriteria objective function, which means that a polynomial time algorithm for optimally solving the problem in its entirety does not exist. There are only a few research investigations that have looked at dual resource constrained scheduling problems but none ever developed a mathematical model to fully characterize the problem. In this dissertation we develop a linear mathematical programming model for this complex dual resource constrained scheduling problem. Already existing methods of operations research such as the branch-and-bound method can solve small problems optimally but they would require an excessive amount of computation time or even incapable of solving problems of industrial size. Since the scheduling decisions must be made quickly, the interest is in developing fast algorithms that are able to identify high quality solutions. For this reason, highly effective meta-search heuristic algorithms based on tabu search are developed in order to quickly solve our scheduling problems. In order to find good quality solutions efficiently, three different heuristics are developed and compared. Tabu search is frequently used in scheduling problems and has shown to produce very good results in complex scheduling problems. In particular, we are aiming to develop three different hybrid search algorithms to compare both their effectiveness and efficiency and identify the best algorithm for dual resource scheduling problems. The tabu search based heuristics are designed to handle the dual resource scheduling problem at two layers merged together, where each layer is a tabu search algorithm itself. One layer is for the machine and the other layer is assigned to the labor resource. In the absence of knowing the actual optimal solutions in industry size problems, another challenge is to assess the quality of the solutions identified by the meta-search heuristics. For that purpose, methods that identify strong lower bounds are proposed. The quality of a solution is then quantified as its percentage deviation from the lower bound. Three different lower bounding methods are proposed here. The first method is logic-based Benders decomposition (LBBD). In this method the problem is decomposed into two parts. In the master problem (MP), we are dealing with a hybrid flowshop with machine as the only constraint. Then after finding the best schedule of jobs on the machines from the MP, the best labor schedule for that machine schedule is found by the subproblem (SP). This leads into developing the benders cut and by feeding the cuts to the MP and resolving the MP, the lower bound is improved. The iterative process between the MP and SP terminates either when the optimal solution is obtained or when the time limit has been reached. The second and third methods are two different variations of iterative selective LP relaxation. In these methods, first all of the binary variables are relaxed in a bound between 0 and 1 and the LP model is solved to generate a lower bound. After that, a selective group of relaxed variables will be set as binary variables and the new model will be solved again to obtain a new lower bound. The methods are different in their variable selection technique. This iterative process is continued until two same lower bounds are obtained or the time limit is reached. Variables are selected in a way that in every iteration the number of binary variables increases and therefore leads to a better or at least the same lower bound found in the previous iteration.

1.2. Outline of the Dissertation

Chapter 2 reviews the literature on dual resource scheduling, unrelated-parallel machine scheduling, flowshop scheduling and hybrid flowshop scheduling problems. In chapter 3, the problem is described in detail, a representative example problem is presented, and the NP-hardness property of the problem is proved. Chapter 4 presents a mixed-integer linear programming (MILP) formulations of the problem. Chapter 5 provides a brief background on the tabu search concept, describes the components and algorithmic structures of the proposed hybrid algorithms, and demonstrates algorithms on the representative example problem. Chapter 6 proposes three lower bounding methods. First for logic-based Benders decomposition, the necessary background is given and master and subproblem formulations are presented. Then iterative selective LP relaxation method is described. Chapter 7 presents a carefully designed computational experiment to test the performance of the proposed meta-heuristics with respect to the proposed lower bounds and/or optimal solutions, and performs statistical analysis to interpret the results. Finally, Chapter 8 concludes the dissertation with the discussion of the results and contributions, and introduces ideas for future research.

2. LITERATURE REVIEW

The flowshop scheduling problem has attracted many researchers since Johnson's paper [2]. In a flowshop, jobs flow through multiple stages in series, where each stage consists of only one machine. In today's competitive environments, some manufacturing industries place additional machines in some stages to increase the capacity or to balance the capacity of the flowshop. The other reason for adding some machines to a stage is due to the increasing demand for customized products, i.e. special packaging, sizes or features. Furthermore, sometimes, when manufacturing industries buy new machines to ensure attaining a specific quality or to produce new products, they do not remove the existing machines to maintain flexibility. This extended layout is usually addressed as hybrid flowshop, flow shop with multiple machines, flexible flowshop, multiprocessor flowshop, or flowshop with parallel machines. By definition this extended layout has series of production stages and at least one stage must have two or more machines operating in parallel. The jobs flow in the same direction through the shop. Each job is processed by one machine in each stage and it must go through one or more stages. Machines in each stage can be identical or unrelated [3]. In the case that processing time for each job on each machine in a stage is the same, the machines are said to be identical. If the processing time for each job on each machine in a stage is different, the machines are unrelated [4]. In this research, we refer to flowshop with identical-parallel machines as flexible flowshop (FFS) and we refer to flowshop with unrelated-parallel machines as hybrid flowshop (HFS). The HFS scheduling problem may be seen as a generalization of two particular types of scheduling problems: the parallel machine scheduling (PMS) problem and the flowshop scheduling (FSS) problem. The key decision of the PMS problem is the allocation of jobs to machines whereas the key decision of FSS is the sequence of jobs through the shop. Hence, once the configuration of the HFS has been designed, the main decisions in the operation of the HFS are to assign and to schedule the jobs to the machines in each stage, i.e. to determine the order in which the jobs are to be processed on the different machines of each stage according to one or several given criteria. The HFS in this research has dual resource constraints and bicriteria objective function. This decision has a large impact on the performance of the HFS and thus it is not surprising that a great research effort has been devoted to this topic [5]. Figure 1 illustrates schematically the relationships between the different machine environments.

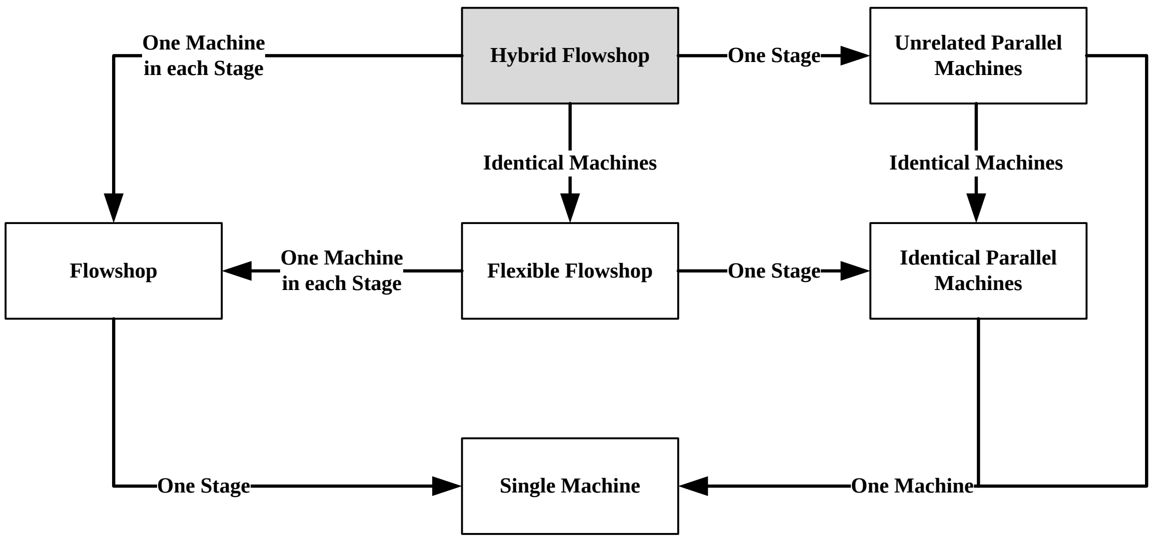

Figure 1. Relationships between different machine environments

2.1. Bicriteria Objective Functions

Supply chain management is becoming an increasingly important issue. One of the most active topics for research in manufacturing over the last 10 years has been supply chain management. Successful companies are those that consider both producer’s and customers’ needs. However, supply chain scheduling has rarely been taken into consideration. The studies on scheduling problems with bicriteria goal attempt to consider the coordination of the producer and customers.

Chen and Vairaktarakis [6] studied an integrated scheduling model of production and distribution operations. In their model, a set of jobs are first processed in a processing facility and then delivered to the customers directly without intermediate inventory. They tried to find a joint schedule of production and distribution with a bicriteria objective function that takes into account both customer service level and total distribution cost. Customer service level is measured by a function of the times when the jobs are delivered to the customers. The distribution cost of a delivery shipment consists of a fixed charge and a variable cost proportional to the total distance of the route taken by the shipment. Eren and Güner [7] proposed a mixed-integer programming model to find the optimum schedule for a single-machine problem with sequence-dependent setup times and they used the same bicriteria objective as in our research to minimize the sum of completion times and to minimize the total tardiness. They did not assume individual weights assigned to each job.

Mansouri et al. [8] introduced two algorithms in a two-machine flowshop with sequence-dependent setup focused on a bicriteria objective of minimizing both setup and makespan where each job is characterized by a pair of attributes that entail setups on each machine. The setup times are sequence-dependent on both machines. Because their objectives have conflict the pareto optimization approach is considered.

Köksalan and Burak Keha [9] considered two different bicriteria objectives on a single machine to minimize the flow time and number of tardy jobs, and to minimize the flow time and maximum earliness. For the first problem, they developed a heuristic that produces an approximately efficient solution and then they developed a genetic algorithm that further improves the approximate solutions. The genetic algorithm was modified for the second problem to match its special structure.

Chauhan et al. [10] believe that successful companies are those that reach at least the two following objectives: reduce their Work-In-Process (WIP) and respond to customer's requirements in real time. To do so, they proposed a model with the objective of minimizing the makespan in order to minimize the WIP cost. They consider customers' time window constraints as customers' satisfaction level. Also, as a result of using the time window constraint in their model, they justify minimizing makespan as a substitute for minimizing WIP.

Chen and Hall [11] considered a bicriteria goal to minimize the total completion time as suppliers saving in terms of WIP cost and to minimize the maximum lateness as it is a measure, which reflects the worst case customer service level.

Chou and Lee [12] developed an integer programming model to solve a two-machine flowshop bicriteria scheduling problem with release dates for the jobs, in which the objective function is to minimize a weighed sum of total flow time and makespan.

2.2. Dual Resources

In the last couple of decades, researchers have addressed the dual resource constrained job shop problem in the stochastic domain using simulation and heuristics [13-18] and by using queuing-theoretic models [19-21]. Although all of these studies dealt with dual resource constraints, they cannot be categorized in the same category as the problem introduced in this research. All of these studies focused on the stochastic characteristics of dual resource constrained job shop and used queuing-theoretic models and simulation techniques for solving the problems. There are only a very few papers that have investigated the dual resource scheduling problem in a deterministic environment while several hundreds of papers have investigated the single resource scheduling problem in a deterministic environment. Adding an additional resource in a deterministic environment increases the complexity of a problem in a way that most researchers prefer not to deal with, through careful analysis and development of a mathematical model. Xu et al. [22] provide a review on recent developments in dual resource constrained scheduling problems with a focus on materials published after 1995 and up to 2009. They considered both stochastic and deterministic scheduling in their review. Lobo et al [23] studied a dual resource constrained job shop with minimization of the maximum job lateness. They considered that machines are organized into groups and each worker is assigned to a specific machine group. They develop a procedure to compute a lower bound using a network-flow formulation and determine the performance of their proposed heuristic which is a search algorithm for finding worker allocation. Lobo at al [24] also extended their previous research and proposed optimality criteria to verify that a given allocation corresponds to a schedule that yields the minimum value of maximum lateness. They also developed a heuristic search algorithm for situations in which the optimality criteria are not satisfied. ElMaraghy et al. [25, 26] proposed a genetic algorithm for finding the optimal/near optimal solution for a deterministic dual resource scheduling problem in a flexible manufacturing system. They considered six different goals for their problem, but none of them dealt with bicriteria scheduling. Different skill levels were considered and the setup time was assumed to be included in the processing time. Chaudhry and Drake [27] also developed a search algorithm based on genetic algorithm for identical-parallel machines with worker assignment. They considered a deterministic setup and processing times and the setup times were assumed to be included in the processing times. They did not assume different skill levels for the workers but the processing times are dependent on the number of workers assigned to a particular machine. Mehravaran and Logendran [28] investigated a non-permutation flowshop scheduling problem with dual resources. They considered different skill levels for labors, however the skill level required for performing both setup and run was assumed to be the same. They developed a mathematical model and a heuristic search algorithm to find solutions for their proposed problem.

2.3. Unrelated-Parallel Machines

In scheduling problems, unrelated-parallel machine scheduling has been widely used in industry. Unrelated-parallel machine scheduling refers to machine arrangements in which a job should pass only through one of the machines which can perform the same function but with different capabilities. Single resource parallel machine scheduling problems have been studied by many researchers.

Chauhan et al. [10] tried to find the optimal schedule of jobs on identical-parallel machines within a specified time window by customers. In their study the machine setup times were disregarded. Suresh and Chaudhuri [29] considered the problem of scheduling jobs on unrelated-parallel machines when machine vacations are specified. Their objective was to minimize the makespan. They did not consider a sequence-dependent setup time and no mathematical model was presented for the problem. None of these researches considered dynamic job release and dynamic machine availability times. Logendran and Subur [30] developed a linear mixed-integer programming (MILP) model and a tabu search-based algorithm to solve an unrelated-parallel machine scheduling problem. They did not assume dual resources and the setup times were considered to be included in the run times to evaluate the job processing times. Logendran et al. [31] developed a tabu search algorithm to optimize a single resource unrelated-parallel machine scheduling problem with sequence-dependent setup times. Their model characteristics were the same as they are in this research problem, but the only resource in their study was the machine and their goal was to minimize the weighted sum of tardiness. Mehravaran and Logendran [32] extended the work by Logendran et al. [31] and developed a linear MIP model with a bicriteria goal but with machine as the only resource. They were the first to address bicriteria scheduling on unrelated-parallel machines. Yaghubian et al. [33] formulated an integer linear program for a dry kiln scheduling problem. The objective of the formulation is to minimize the maximum tardiness of *n* independent jobs on *m* non-identical parallel machines. Azizoglu and Kirca [34] considered unrelated-parallel machine scheduling problems that involve the minimization of regular total cost functions. Ying et al. [35] considered a problem of scheduling jobs on unrelated-parallel machines with machine-dependent and job sequence-dependent setup times. They used a restricted simulated annealing algorithm to minimize the makespan. Tavakkoli-Moghaddam et al. [36] presented a two-level bi-objective mixed-integer programming model for scheduling unrelated-parallel machines. They divided the model into two parts. The first part minimizes the number of tardy jobs and the result from the first part is inserted to the second part to minimize the total completion time of all the jobs. They considered non identical due dates and ready times with sequence-dependent setup times and proposed a genetic algorithm to solve the problem. Fanjul-Peyro and Ruiz [37] developed a set of simple greedy local search to minimize the makespan on an unrelated parallel machine scheduling problem. Their problem ignored the setup times and only considered processing time. Rocha et al. [38] proposed a scheduling problem with unrelated-parallel machines, sequence and machine-dependent setup times, due dates and weighted jobs. An upper bound was found by developing a generic branch-and-bound algorithm using a greedy random adaptive search procedure as an initialization procedure. Results from their heuristic were compared with MILP models solved by CPLEX.

Every stage of the HFS problem in this dissertation can have unrelated or identical parallel machines structure. Each stage has similar characteristics as the previous studies unless in this research we are dealing with labor in addition to the machines.

2.4. Flowshop

Flowshop is one of the most commonly used arrangements in production scheduling and finding an optimal schedule of jobs in a flowshop has been a great interest for researchers and practitioners alike. Flowshop is an arrangement of machines in which n jobs have to be processed on m machines and every job has to be processed at most once on a machine, and each machine can only process one job at a time. Processing of a job must be completed on the current machine before processing of the job is started on the succeeding machine. Assembly lines are one of the most common examples of flowshop. Other applications of this arrangement can be found in electronics manufacturing, and space shuttle processing [39]. It should be noted that the problem considered in this research is a hybrid-flowshop. In finding the optimal schedule for a flowshop, two main approaches are considered, permutation and non-permutation schedules. In permutation schedules, the sequence of jobs is assumed to remain the same on all machines, whereas in non-permutation schedules the sequence can be different. Considering non-permutation flowshop schedules are more challenging; there are $n!$ permutation sequences compared to $(n!)^m$ non-permutation sequences. Even when permutation sequences are considered, finding the optimal solution efficiently from among $n!$

sequences can be very challenging, let alone finding the optimal solution from among $(n!)^m$ non-permutation sequences.

Ruiz and Maroto [40] evaluated 25 different heuristics for finding permutation schedules in order to minimize the makespan. Srikar and Ghosh [41] developed a mixed-integer program for solving a permutation flowshop with sequence-dependent setup time with either makespan or total completion time objective. Liao and Huang [42] developed a non-permutation schedule for minimizing the tardiness. They modeled the problem with three different mathematical models and used two tabu search-based algorithms for finding the flowshop schedule, but they only use a processing time and do not separate the setup and run times in their flowshop. Ying [35] developed a greedy heuristic for solving non-permutation flowshop with the goal of minimizing the makespan. Aggoune and Portman [43] addressed the flow shop scheduling problem with limited machine availability to minimize the makespan. Mehravaran and Logendran [44] developed a mathematical model and search algorithm based on tabu search to solve a permutation and non-permutation flowshop with a bicriteria objective. But all of them considered machine as the *only* resource.

Flowshop scheduling problems are a special case of the HFS scheduling problems addressed in this research with only one machine at each stage. Although the previous research summarized in this subsection has lots of similarities to our problem such as sequence-dependent setup, bicriteria objective function and dynamic job releases and machine availability times, they lack two distinctive and very important properties. The first one, as mentioned above, is multiple machines at one or more stages and the second and perhaps the most important distinction is the presence of labor as the second constrained resource.

2.5. Hybrid Flowshop

A Hybrid Flow Shop (HFS) consists of series of production stages, each of which has several machines operating in parallel. Some stages may have only one machine, but at least one stage must have multiple (two or more) machines. The flow of jobs through the shop is unidirectional. Each job is processed by one machine in each stage and it must go through one or more stages [45]. Machines in each stage can be identical, uniform or unrelated. In fact, HFS is often found in the electronic manufacturing environment such as IC packaging and PCB fabrication [3]. Ribas et al [5] provided a review on HFS scheduling problem with the main focus on papers from 1995 on and Lin and Zhang [3] provided a review on hybrid flowshop scheduling papers prior to 1999.

Most of the researchers studied hybrid flowshop with identical machines or flexible flowshop. Jin et al [46] considered the multistage hybrid flowshop scheduling problem, in which each stage consists of parallel-identical machines. The problem is to determine a schedule that minimizes the makespan for a given set of jobs over a finite planning horizon. They first used a series of new global lower bounds to estimate the minimum makespan and then developed new meta-heuristic algorithms based on simulated annealing. Tang et al [47] investigated the problem of scheduling n jobs in t stage hybrid flowshop with parallel-identical machines at each stage. The objective is to find a schedule that minimizes the sum of weighted completion times of the jobs. In this paper, an integer programming formulation is constructed for the problem and a new Lagrangian relaxation algorithm is presented. Portman et al [48] developed optimal methods for solving t stage hybrid flowshop scheduling problem. They improved the lower bound created for the same problem by Brah and Hunsucker [49] and introduced several heuristics to compute an initial upper bound and genetic algorithm (GA) to improve the value of the upper bound during the search. Lee and Kim [50] considered a two-stage hybrid flowshop-scheduling problem with the objective of minimizing total tardiness of jobs. In the hybrid flowshop, there is one machine at the first stage and multiple identical-parallel machines at the second stage. They developed a branch-and-bound algorithm that can find optimal solutions for problems with up to 15 jobs in a reasonable amount of central processing unit time. Haouari and M'Hallah [51] considered a two-stage hybrid flowshop problem with several parallel machines in each stage and *n* jobs to be processed on at most one machine per stage. The objective is to minimize the maximum completion time. They developed two two-phase methods based on Simulated Annealing and Tabu Search. By comparing the results from heuristics with a newly derived lower bound, they showed the superiority of the derived lower bound and the efficiency and effectiveness of the proposed heuristic. Alaykyran et al [52] used ant colony optimization method to solve hybrid flow shop problems with the objective of minimizing the makespan. In a hybrid flow shop, machines are arranged into *t* stages in series and each stage has identical machines in parallel. Allaoui and Artiba [53] investigated the two-stage hybrid flow shop scheduling problem with only one machine in the first stage and *m* identical parallel machines in the second stage to minimize the makespan. They considered that each machine is subject to at most one unavailability period and the start time and the end time of each period are known in advance. Zandieh et al [54] studied hybrid flow shop scheduling problems with identical parallel machines in each stage. The setup time is considered to be sequence-dependent. They used an immune algorithm to tackle this problem. Choi et al [55] focused on a hybrid flowshop scheduling problem, in which there are serial stages, each with identical-parallel machines. They developed heuristic algorithms with the objective of minimizing total tardiness. Gupta and Tunk [56] considered the two-stage hybrid flowshop problem with the objective of minimizing the total number of tardy jobs. The first stage contains only one machine and the second stage contains *m* identical-parallel machines. They developed several heuristic algorithms to find optimal or near optimal schedule. Haouari and Hidri [57] considered a hybrid flowshop scheduling problem to minimize the makespan in a serial multiple-stage manufacturing system, where each stage consists of parallel identical machines. They proposed a valid lower bound in their paper. Janiak et al [58] studied the flow shop scheduling problem with identical-parallel machines at each stage. The goal is to minimize the cost of the total weighted earliness, the total weighted tardiness and the total weighted waiting time. They developed three algorithms based on Tabu Search and Simulated Annealing techniques to solve the problem. Moursli and Pochet [59] introduced a branch-and-bound algorithm for the hybrid flowshop scheduling problem with identical-parallel machines in each stage to minimize makespan. Several heuristics are developed to compute upper bounds. Lower bounds are based upon the single-stage subproblem relaxation. Naderi et al [60] addressed the problem of scheduling hybrid flowshop with identical-parallel machines in each stage where the setup times are sequence-dependent to minimize makespan and maximum tardiness. They developed an algorithm based on simulated annealing techniques to solve the problem. Nishi et al [61] developed a new Lagrangian relaxation method for solving the hybrid flowshop scheduling problem with identical-parallel machines in each stage to minimize the total weighted tardiness. Riane et al [62] treated a problem of scheduling *n* jobs on a three stage hybrid flowshop with one machine in the first and third stages and two identical machines in stage two. The objective is to minimize the makespan. Tseng and Liao [63] addressed a hybrid flowshop scheduling problem with identical-parallel machines. They developed an algorithm based on particle swarm optimization, a novel meta-heuristic inspired by the flocking behavior of birds. Wang and Tang [64] investigated a hybrid flowshop scheduling with identical-parallel machines in each stage and with finite intermediate buffers. The objective function is to minimize the sum of weighted completion time and the authors tried to solve the problem with a tabu search heuristic. Ying [65] proposed a simple iterative greedy heuristic to minimize makespan in a multistage hybrid flowshop with identical-parallel machines. Ying and Lin [66] considered the multistage hybrid flowshop problem with identical-parallel machines. They developed a novel ant colony system heuristic to solve the problem and compared the results with genetic algorithm and tabu search.

Although HFS is more generalized than FFS only few researchers considered HFS. Ruiz and Maroto [67] developed a genetic algorithm to solve a HFS with unrelated-parallel machines at each stage, sequence-dependent setup times and machine eligibility. The goal is to minimize the makespan. Bertel and Billaut [68] considered a hybrid flow shop with unrelated-parallel machines in each stage and recirculation in order to minimize the weighted number of tardy jobs. They developed an integer linear programming formulation, a lower bound, a greedy algorithm and a genetic algorithm to solve the problem. Low et al. [69] addressed a two-stage hybrid flowshop scheduling problem with unrelated alternative machines. The problem has *m* unrelated alternative machines at the first machine center, followed by a single machine at the second center. The objective is to minimize the makespan. Yaurima et al [70] presented a genetic algorithm for solving the hybrid flowshop with unrelated-parallel machines, sequence-dependent setup time, availability constraints, and limited buffers.

As stated above there are many works on FFS, but only few works focus on HFS. In this research, we are emphasizing to optimize a HFS with unrelated-parallel machines at each stage and sequence-dependent setup times like in most of the previous works. In addition, our research deals with bicriteria objective function, and dynamic machine availability and job releases times. However, the main characteristic of this dissertation that has never been studied before is the dual resource constraint property. We have developed a linear mixed-integer programming model, two methods of lower bounding, and two different heuristics, one based on tabu search and one based on genetic algorithm, to solve the problem. To the best of our knowledge, there is no prior work on dual resource scheduling that deals with this problem as comprehensively as our research, reported in this dissertation.

3. PROBLEM DESCRIPTION

This dissertation addresses a hybrid-flow shop scheduling problem in the presence of dual resource constraints. Unlike the traditional scheduling problem, the manpower is constrained in addition to the machines. Different manpower skills exist for processing the jobs in all stages. The skill level required to perform the setup could be different from that needed to perform the run. The setup time is assumed to be sequence-dependent, and job release times and machine availability times are considered dynamic. It means that at the start of the planning horizon not all jobs may be available for processing and some machines may still be processing jobs released in the previous planning horizon. Although job release times and machine availability times are considered to be dynamic, the problem still remains in the off-line scheduling domain. In most industry-scheduling problems, it is not practically possible to have all jobs and machines ready and available at the start of the planning horizon. By considering this property, we ensure that the proposed problem is investigated with settings typically observed in real problems in industry. Also machine skipping is allowed. The problem schematic is shown in Figure 2.

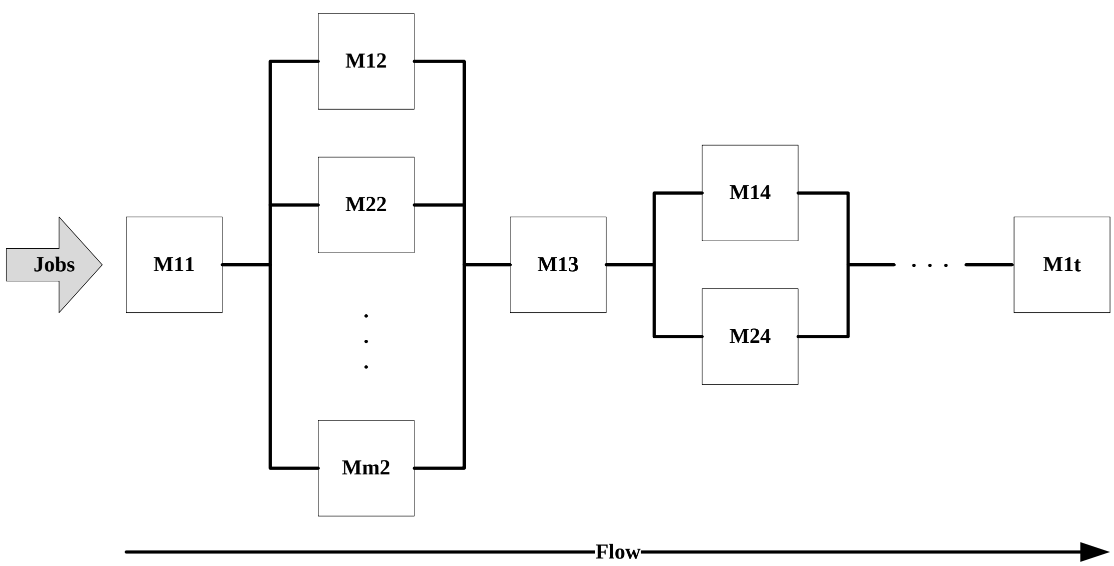

Figure 2. Problem schematic

In this research, we propose a bicriteria objective in order to minimize both the total completion time and total tardiness. Trying to optimize two contrasting objectives enables incorporating the coordination that must be maintained between the producer and the customers in a supply chain. We attempt to find a schedule in which the producer's work-in-process inventory (WIP) cost is minimized, while the customers' service level is maximized. By considering a bicriteria goal for this problem we are trying to optimize two contrasting objectives. Liao and Huang [42] stated that the complexity of minimizing the total completion time and total tardiness (separately) is more than minimizing the makespan. Minimizing the completion time is desirable for the producer so they can minimize their work-in-process inventory; however, Armentano and Ronconi [71] recognized that lots of manufacturers are now more interested in meeting the customers' due dates and maximizing the customers' service level by minimizing the tardiness. In our research to make the situation even more realistic, a weight is assigned to each job which shows the importance of those jobs in terms of WIP cost and service level.

The bicriteria objective is normalized by the weights assigned to each criterion. Defining different normalized weights creates different scenarios. The scenarios are identified to convert the completion time and tardiness to their relevant costs. In reality, evaluating the cost of WIP or the cost of not meeting customers' due date is hard or even impossible and so in most of the previous research, the corresponding total completion time or tardiness is used. This is a perfectly valid assumption as the unit cost of WIP and a job being tardy is assumed to be fixed and does not have any effect on the objective function when considering a single-criterion objective. However, this is not a true statement in bicriteria scheduling as the unit cost of WIP and jobs being tardy is not the same. As stated before, evaluating these unit costs are really challenging. Different scenarios that contain different normalized weights will help the manufactures to find a situation (scenario) that meets their needs and describes their cost best. In this research a traditional and the most common method of "weighted aggregating function" is used to deal with the bicriteria objective instead of the Pareto optimization. Other studies [7, 32, 44] have also used weighted aggregating function. Usually, Pareto optimization is used when some of the criteria are in conflict [72]. The satisfaction of both completion time and tardiness objectives in this research moves in the same direction and hence the objectives are not in conflict. Because of this, it is really complex to form the Pareto optimal frontier and it would not serve the purpose of the research.

3.1. A Representative Example

The representative example is a small problem with 3 jobs and 3 stages. There are two unrelated-parallel machines in stage 1 (M11 and M21), one machine in stage 2 (M12), and one machine in stage 3 (M13). There are two units of labor, one labor has skill 1 and the other has skill 2. The hybrid-flow shop is shown in Figure 3.

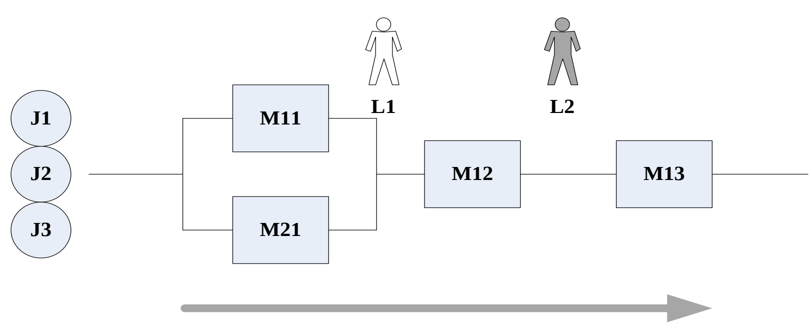

Figure 3. Schematic of a small example

Job 1 (J1) goes through stage 1, stage 2 and stage 3. Both machines (M11 and M21) in stage 1 are capable of processing J1, however as M11 and M21 are unrelated the processing time for J1 is different on M11 and M21. If J1 is processed on M11 the run time is 36 and it needs labor with skill level 2 to perform its setup and labor with skill level 1 to perform its run. However, if J1 is processed on M21 the run time is 12 and it needs labor with skill level 1 to perform both setup and run. Job 2 (J2) has a process on stages 1 and 3. J2 skips the second stage and only M11 is capable of performing it in stage 1. Job 3 (J3) has a process on stages 1 and 2 and is skipping stage 3. Table 3.1 shows the job run times on different machines and labor skills required for every operation. Zero processing time means that the machine is not capable of processing that job.

Table 3.1 Assignment of jobs to machines and labor with their run time

			J1	J2	J3
Stage 1	M11	Run time	36	41	25
		Setup Skill	2	1	2
		Run Skill	1	1	1
	M21	Run time	12	0	15
		Setup Skill	1	0	2
		Run Skill	1	0	1
Stage 2	M12	Run time	12	0	17
		Setup Skill	2	0	2
		Run Skill	1	0	1
Stage 3	M13	Run time	14	30	0
		Setup Skill	1	2	0
		Run Skill	1	2	0

The setup time on machines for every job is sequence-dependent which means the setup time of a machine for a job is dependent on the previous job processed on the same machine. Table 3.2 shows the sequence-dependent setup time of machines for every job. For example, the setup time for J1 on M11 is 29 if J1 is processed immediately after J2, or it is 31 if J1 is processed immediately after J3, or is 4 if J1 is the first job that is processed in this planning horizon. Ref is the reference job and it is the last job that was processed on M11 in the previous planning horizon.

Table 3.2 Sequence-dependent setup time

M11			
K \ J	1	2	3
1	0	9	15
2	29	0	20
3	31	34	0
Ref	4	12	32

M21			
K \ J	1	2	3
1	0	0	38
2	0	0	0
3	37	0	0
Ref	16	0	17

M12			
K \ J	1	2	3
1	0	0	30
2	0	0	0
3	15	0	0
Ref	12	0	36

M13			
K \ J	1	2	3
1	0	38	0
2	12	0	0
3	0	0	0
Ref	2	21	0

There is a weight assigned to each job which shows the importance of that job in terms of work-in-process inventory and service level. The scenario used for this problem is 70%-30%, which means that 70% of the objective function value is evaluated from total weighted completion time and 30% of it is evaluated from total weighted tardiness. Jobs have dynamic release times and machines have dynamic availability times. Also jobs have to be delivered to customers by their due date. Table 3.3 shows the jobs' weights, release times and due dates and Table 3.4 shows machine availability times.

Table 3.3 Job information

Job	J1	J2	J3
Weight	1	2	2
Release Time	15	4	31
Due date	139	205	109

Table 3.4 Machine availability time

Machines	M11	M21	M12	M13
Availability Times	39	28	82	104

3.2. Complexity of the Research Problem

This section shows that the problem is NP-hard in the strong sense. There are a number of techniques to prove that a problem is a member of a certain complexity class. The method used here is proof by restriction [73]. An NP-hardness proof by restriction for a given problem P consists of simply showing that P contains a known NP-hard problem Q as a special case. The main point in this method lies in the specification of the additional restrictions to be placed on the instances of P so that the resulting restricted problem will be identical to Q.

Theorem 3.2.1. *Hybrid flowshop scheduling problem with machine as the only resource and minimization of sum of completion time is NP-hard in the strong sense.*

Proof. Let P be the hybrid flowshop scheduling problem with machine as the only resource and minimization of sum of completion time. Construct problem Q as a two machine flowshop scheduling problem with a single machine in each stage and the objective of minimizing the sum of completion time. Clearly, problem Q is a special case of problem P (even when problem P involves more than two machines, since all the setup and run times on the machines other than the first two can be restricted to be 0 in problem P). Observe that problem Q is equivalent to the two-machine flowshop scheduling problem with the objective of minimizing the sum of completion time, which has been shown to be NP-hard in the strong sense [74]. It follows that problem P is NP-hard in the strong sense.

Theorem 3.2.2. *Hybrid flowshop scheduling problem with machine as the only resource and minimization of sum of tardiness is NP-hard in the strong sense.*

Proof. Let P be the hybrid flowshop scheduling problem with machine as the only resource and minimization of sum of tardiness. Construct problem Q as a single machine scheduling problem and the objective of minimizing the sum of tardiness. Clearly, problem Q is a special case of problem P (even when problem P involves more than one machine, since all the setup and run times on the machines other than the first one can be restricted to be 0 in problem P). Observe that problem Q is equivalent to the single machine scheduling problem with the objective of minimizing the sum of tardiness, which has been shown to be NP-hard in the strong sense [75]. It follows that problem P is NP-hard in the strong sense.

Theorem 3.2.3. *Hybrid flowshop scheduling problem with machine as the only resource and bicriteria objective function of minimization of weighted sum of weighted completion time and weighted tardiness is NP-hard in the strong sense.*

Proof. Let P be the hybrid flowshop scheduling problem with machine as the only resource and bicriteria objective function of minimization of weighted sum of weighted completion time and weighted tardiness. Construct problem Q as a hybrid flowshop scheduling problem and minimization of sum of completion time and construct problem Q’ as a hybrid flowshop scheduling problem and minimization of sum of tardiness. Clearly, problem Q and Q’ are special cases of problem P (problem P can be either Q by setting the weight of individual job’s equal to one and setting the sum of weighted tardiness’s weight equal to zero or Q’ by setting the weight of individual job’s equal to one and setting the sum of weighted completion time’s weight equal to zero). Problems Q and Q’ are proved to be NP-hard in strong sense in Theorems 3.2.1. and 3.2.2., respectively. It follows that problem P is NP-hard in the strong sense.

Theorem 3.2.4. *Unrelated-parallel machine scheduling problem with bicriteria objective function of minimization of weighted sum of weighted completion time and weighted tardiness is NP-hard in the strong sense.*

Proof. Let P be the unrelated-parallel machine scheduling problem with bicriteria objective function of minimization of weighted sum of weighted completion time and weighted tardiness. Construct problem Q as an unrelated-parallel machine scheduling problem with bicriteria objective function of minimization of sum of completion time. Problem Q has been shown to be NP-hard [76]. Construct problem Q’ as a single machine scheduling problem and the objective of minimizing the sum of tardiness. Problem Q’ also has been shown to be NP-hard in the strong sense [75]. Clearly, problem Q and Q’ are special cases of problem P (problem P can be either Q by setting the weight of individual job’s equal to one and setting the sum of weighted tardiness’s weight equal to zero or Q’ by setting the weight of individual job’s equal to one and setting the sum of weighted completion time’s weight equal to zero and having only one single machine in the system). Both problems Q and Q’ are proved to be NP-hard and so P is NP-hard in the strong sense.

Theorem 3.2.5. *Hybrid flowshop scheduling problem with dual resources and bicriteria objective function of minimization of weighted sum of weighted completion time and weighted tardiness is NP-hard in the strong sense.*

To prove that hybrid flowshop scheduling problem with dual resources and bicriteria objective function is NP-hard, we should prove that hybrid flowshop scheduling problem and bicriteria objective function with each of its resources is NP-hard. Hybrid flowshop scheduling problem with machine as the only resource and bicriteria objective function was proved to be NP-hard in Theorem 3.2.3. When labor is the only resource in hybrid flowshop scheduling problem, it does not remain a hybrid flowshop anymore. Hybrid flowshop is an arrangement made by machines in a job shop, however, when the machine resource is omitted, this arrangement will not be respected anymore. In this case the hybrid flowshop scheduling problem with labor as the only resource and bicriteria objective function, is equivalent to unrelated-parallel machine scheduling problem with bicriteria objective function. The problem has *l* machines (equivalent to total number of labor) and *n*T*2* jobs (equivalent to total number of jobs*total number of stages*[setup/run]).

Proof. Let P be the hybrid flowshop scheduling problem with dual resources and bicriteria objective function of minimization of weighted sum of weighted completion time and weighted tardiness. Construct problem Q as a hybrid flowshop scheduling problem with machine as the only resource and bicriteria objective function of minimization of weighted sum of weighted completion time and weighted tardiness and construct problem Q' as an unrelated-parallel machine scheduling problem with bicriteria objective function of minimization of weighted sum of weighted completion time and weighted tardiness. Clearly, problem Q and Q' are special cases of problem P (problem P can be either Q when machine is the only resource or Q' when labor is the only resource). Problems Q and Q' are proved to be NP-hard in strong sense in Theorems 3.2.3. and 3.2.4., respectively. It follows that problem P is NP-hard in the strong sense.

4. MATHEMATICAL MODEL

The problem has $j \in \{1,2,..,n\}$ jobs. Each job should pass through a hybrid flowshop with $t \in \{1,2,\dots,T\}$ stages. There are $i \in \{1,2,..,m_t\}$ unrelated parallel machines in each stage where job *j* has to pass only through one of them with a run time of p_{ijt} and it has a setup time of s_{ikjt}, where job *k* is the job being processed before job *j* on machine *i* at stage *t*. If job *j* is the first job to be processed on machine *i* at stage *t*, then *k* = 0, which is called the reference job. Realistically, the reference job is the last job from the previous planning horizon to be processed on that machine. Each job *j* has a weight of w_j, release time of r_j and due date of d_j. Also each machine *i* at stage *t* is available at time a_{it}. If job *j* skips stage *t*, $o_{jt} = 0$, and if job *j* has an operation on stage *t*, then $o_{jt} = 1$.

Also both setup and run of each job *j* should be assigned to each labor $l \in 1,2,\dots,p$. But there is an assignment criteria that should be considered, $g_{ijlt} = 1$ if labor *l* can process job *j* setup on machine *i* at stage *t*, and $g'_{ijlt} = 1$ if labor *l* can process job *j* run on machine *i* at stage *t*.

α is the weight assigned to the producer and attributed to the total completion time ($0 \le \alpha \le 1$) and β is the weight assigned to the customer and attributed to the total tardiness, and $\alpha + \beta = 1$, meaning the weights are normalized. Different values of α and β defines different scenarios in the problem.

Variables:

SB_{jt}= setup start time of job *j* at stage *t*

SC_{jt} = setup completion time of job *j* at stage *t*

RB_{jt} = run start time of job *j* at stage *t*

RC_{jt} = run completion time of job *j* at stage *t*

TD_j = tardiness of job *j*

$$x_{ijt} = \begin{cases} 1, \text{if job } j \text{ is assigned to machine } i \text{ at stage } t \\ 0, \text{otherwise} \end{cases}$$

$$y_{ikjt} = \begin{cases} 1, \text{if job } k \text{ scheduled immediately before job } j \text{ on machine } i \text{ at stage } t \\ 0, \text{otherwise} \end{cases}$$

$$z_{ktjt'l} = \begin{cases} 1, \text{if job } k \text{ setup at stage } t \text{ scheduled anytime before job } j \text{ setup at stage } t' \text{on labor } l \\ 0, \text{otherwise} \end{cases}$$

$$z'_{ktjt'l} = \begin{cases} 1, \text{if job } k \text{ setup at stage } t \text{ scheduled anytime before job } j \text{ run at stage } t' \text{on labor } l \\ 0, \text{otherwise} \end{cases}$$

$$q_{ktjt'l} = \begin{cases} 1, \text{if job } k \text{ run at stage } t \text{ scheduled anytime before job } j \text{ setup at stage } t' \text{ on labor } l \\ 0, \text{otherwise} \end{cases}$$

$$q'_{ktjt'l} = \begin{cases} 1, \text{if job } k \text{ run at stage } t \text{ scheduled anytime before job } j \text{ run at stage } t' \text{ on labor } l \\ 0, \text{otherwise} \end{cases}$$

$$u_{jlt} = \begin{cases} 1, \text{if job } j \text{ uses labor } l \text{ for its setup at stage } t \\ 0, \text{otherwise} \end{cases}$$

$$v_{jlt} = \begin{cases} 1, \text{if job } j \text{ uses labor } l \text{ for its run at stage } t \\ 0, \text{otherwise} \end{cases}$$

The objective function is to minimize the weighted sum of weighted completion time and weighted tardiness:

$$\min \quad z = \alpha \sum_{j=1}^{n} w_j RC_{jT} + \beta \sum_{j=1}^{n} w_j TD_j \quad (1)$$

Hybrid flowshop with dual resource scheduling can be formulated into two parts. The first part contains machine only constraints and the second part contains labor only constraints.

4.1. Machine Only Constraints

$$\sum_{i=1}^{m_t} x_{ijt} = o_{jt}, \ j = 1,2,\dots,n \ \ t = 1,2,\dots,T. \ (2)$$

Constraints (2) assign jobs to the machines. These constraints ensure that each job is processed only on one machine.

$SB_{jt} \geq a_{it}x_{ijt}, \quad i = 1,2,\dots,m_t.\ j = 1,2,\dots,n\ \ t = 1,2,\dots,T.$ (3)

Constraints (3) evaluate the setup start time. At the earliest time, job setup time can start after the machine performing on that job is available.

$SC_{jt} \geq SB_{jt} + s_{i0jt}y_{i0jt}, \quad i = 1,2,\dots,m_t.\ j = 1,2,\dots,n\ \ t = 1,2,\dots,T.$ (4)

$SC_{jt} - SB_{jt} + M\left(1 - y_{ikjt}\right) \geq s_{ikjt}x_{ijt},\ i = 1,\dots,m_t.\ j,k = 1,\dots,n. k < j\ \ t = 1,2,\dots,T.$ (5)

$SC_{kt} - SB_{kt} + M(1 - y_{ijkt}) \geq s_{ijkt}x_{ikt},\ i = 1,\dots,m_t.\ j,k = 1,\dots,n. k < j\ \ t = 1,2,\dots,T.$ (6)

Constraints (4), (5) and (6) calculate the setup completion time. Setup completion time is equal to setup start time plus the setup time. Constraints (4) evaluate setup completion time when job *j* is the first job to be scheduled at stage *t*, while constraints (5) and (6) evaluate setup completion time when job *j* is not the first job to be scheduled at stage *t*. Constraints (5) ensure that job *j* setup completion time at stage *t* is at least equal to job *j* setup start time plus job *j* setup time on machine *i* at stage *t* when job *k* is scheduled immediately before job *j* on machine *i* at stage *t* ($y_{ikjt} = 1$). Likewise, constraints (6) state the job *k* setup completion time at stage *t*.

$SB_{jt} - RC_{kt} + M\left(1 - y_{ikjt}\right) \geq 0,\ i = 1,\dots,m_t.\ j,k = 1,\dots,n. k < j\ \ t = 1,2,\dots,T.$ (7)

$SB_{kt} - RC_{jt} + M\left(1 - y_{ijkt}\right) \geq 0,\ i = 1,\dots,m_t.\ j,k = 1,\dots,n. k < j\ \ t = 1,2,\dots,T.$ (8)

$\sum_{j=1,k\neq j}^{n} y_{ikjt} \leq 1,\ \ i = 1,2,\dots,m_t,\ k = 0,1,2,\dots,n\ \ t = 1,2,\dots,T.$ (9)

$\sum_{k=0,k\neq j}^{n} y_{ikjt} = x_{ijt},\ \ i = 1,2,\dots,m_t,\ j = 1,2,\dots,n\ \ t = 1,2,\dots,T.$ (10)

$y_{ikjt} \leq x_{ikt}, \quad i = 1,2,\dots,m_t,\ j,k = 1,2,\dots,n, k \neq j\ \ t = 1,2,\dots,T.$ (11)

Constraints (7), (8), (9), (10), and (11) determine the sequence of jobs to the machines. Constraints (7) and (8) jointly ensure that two jobs cannot be processed on a same machine at the same time, and also calculate setup start time when it is not the first job at the stage. Constraints (7) ensure that job *j* setup start time at stage *t* is at least equal to job k run completion time at stage t when job k is scheduled immediately before job j on machine i at stage t ($y_{ikjt} = 1$). Likewise, constraints (8) state that job *k* setup start time at stage t is at least equal to job j run completion time at stage t if job j is scheduled immediately before k on machine i at stage t ($y_{ijkt} = 1$). Constraints (9) guarantee that a job cannot be succeeded by more than one job. Constraints (10) ensure that a job is preceded by one job. In constraints (11), we make sure that job *j* cannot succeed job *k* on machine *i* at stage *t* unless job *k* is assigned to machine *i* at stage *t*.

$$RB_{jt} \geq SC_{jt}, i = 1,2,\dots,m_t.\ j = 1,2,\dots,n\ \ t = 1,2,\dots,T. \quad (12)$$

$$RB_{jt} \geq r_j x_{ijt},\ i = 1,2,\dots,m_t.\ \ j = 1,2,\dots,n\ \ t = 1,2,\dots,T. \quad (13)$$

Constraints (12) and (13) evaluate run start time. Job run time can start as soon as the setup is completed and the job is available.

$$RC_{jt} \geq RB_{jt} + p_{ijt} x_{ijt},\quad i = 1,2,\dots,m_t.\ j = 1,2,\dots,n\ \ t = 1,2,\dots,T. \quad (14)$$

Constraints (14) calculate run completion time. Run completion time is equal to run start time plus the run time.

$$RB_{jt} \geq RC_{jt-1},\ \ j = 1,2,\dots,n\ \ t = 2,3,\dots,T. \quad (15)$$

Constraints (15) show the stage connection. Run time of job j at stage t can start only after the run of job *j* is completed in the previous stage (*t-1*).

$$TD_j \geq RC_{jT} - d_j,\qquad j = 1,2,\dots,n. \quad (16)$$

Job tardiness is evaluated by constraints (16).

4.2. Labor Only Constraints

$\sum_{l=1}^{p} u_{jlt} = o_{jt}, \;\; j = 1,2, \dots, n \quad t = 1,2, \dots, T.$ (17)

$\sum_{l=1}^{p} v_{jlt} = o_{jt}, \;\; j = 1,2, \dots, n \quad t = 1,2, \dots, T.$ (18)

$u_{jlt} \leq \sum_{i=1}^{m_t} g_{ijlt}.x_{ijt}\,, \;\; l = 1,2,..,p, j = 1,2, \dots, n, t = 1,2, \dots, T.$ (19)

$v_{jlt} \leq \sum_{i=1}^{m_t} g'_{ijlt}.x_{ijt}\,, \;\; l = 1,2,..,p, j = 1,2, \dots, n, t = 1,2, \dots, T.$ (20)

Constraints (17), (18), (19), and (20) assign jobs to labor. Constraints (17) ensure that each job setup is processed only by one labor, and likewise, constraints (18) ensure that each job run is processed only by one labor. Constraints (19) and (20) guarantee that a job *j* (setup or run) assigned to machine *i* at stage *t* ($x_{ijt} = 1$) can be processed only by an eligible labor *l*. Labor *l* is eligible when it can process job *j* on machine *i* at stage *t* and therefore $g_{ijlt} = 1$ for setup and $g'_{ijlt} = 1$ for run.

$SB_{jt'} - SC_{kt} + M(1 - z_{ktjt'l}) + M(2 - u_{jlt'} - u_{klt}) \geq 0,$ (21)

$SB_{kt} - SC_{jt'} + Mz_{ktjt'l} + M(2 - u_{jlt'} - u_{klt}) \geq 0,$ (22)

$RB_{jt'} - SC_{kt} + M(1 - z'_{ktjt'l}) + M(2 - v_{jlt'} - u_{klt}) \geq 0,$ (23)

$SB_{kt} - RC_{jt'} + Mz'_{ktjt'l} + M(2 - v_{jlt'} - u_{klt}) \geq 0,$(24)

$SB_{jt'} - RC_{kt} + M(1 - q_{ktjt'l}) + M(2 - u_{jlt'} - v_{klt}) \geq 0,$ (25)

$RB_{kt} - SC_{jt'} + Mq_{ktjt'l} + M(2 - u_{jlt'} - v_{klt}) \geq 0,$ (26)

$RB_{jt'} - RC_{kt} + M(1 - q'_{ktjt'l}) + M(2 - v_{jlt'} - v_{klt}) \geq 0,$ (27)

$RB_{kt} - RC_{jt'} + Mq'_{ktjt'l} + M(2 - v_{jlt'} - v_{klt}) \geq 0,$ (28)

$l = 1, \dots, p, j, k = 1, \dots, n, t, t' = 1,2, \dots, T, (k < j \; for \; \forall t, t') \cup (k = j \; for \; t < t').$

Constraints (21), (22), (23), (24), (25), (26), (27), and (28) determine the sequence of jobs on labor. These constraints work in pairs. Each pair jointly ensures that two jobs cannot be processed using the same labor at the same time. For instance, constraints (21) and (22) are the first pair. Constraints (21) state that in case both jobs *k* setup at stage *t* and *j* setup at stage *t'* are assigned to labor *l* ($u_{klt} = 1, u_{jlt'} = 1$) and job *k* setup at stage *t* is performed before job *j* setup at stage *t'* by labor *l* ($z_{ktjt'l} = 1$), setup start time of job *j* at stage *t'* has to be greater than or equal to the setup completion time of job *k* at stage *t*. Likewise, constraints (22) evaluate the setup start time of job *k* at stage *t* in case job *j* setup at stage *t'* precedes job *k* setup at stage *t* ($z_{ktjt'l} = 0$). Other pairs work exactly the same way using the same mechanism. Constraints (23) and (24) deal with job *k* setup at stage *t* and job *j* run at stage *t'*. So constraints (23) evaluate the run start time of job *j* at stage *t'* in case job *k* setup at stage *t* precedes job *j* run at stage *t'* (${z'}_{ktjt'l} = 1$) and constraints (24) evaluate the setup start time of job *k* at stage *t* in case job *j* run at stage *t'* precedes job *k* setup at stage *t* (${z'}_{ktjt'l} = 0$). Constraints (25) and (26) deal with job *k* run at stage *t* and job *j* setup at stage *t'*. So constraints (25) evaluate the setup start time of job *j* at stage *t'* in case job *k* run at stage *t* precedes job *j* setup at stage *t'* ($q_{ktjt'l} = 1$) and constraints (26) evaluate the run start time of job *k* at stage *t* in case job *j* setup at stage *t'* precedes job *k* run at stage *t* ($q_{ktjt'l} = 0$). Constraints (27) and (28) deal with job *k* run at stage *t* and job *j* run at stage *t'*. So constraints (27) evaluate the run start time of job *j* at stage *t'* in case job *k* run at stage *t* precedes job *j* run at stage *t'* (${q'}_{ktjt'l} = 1$) and constraints (28) evaluate the run start time of job *k* at stage *t* in case job *j* run at stage *t'* precedes job *k* run at stage *t* (${q'}_{ktjt'l} = 0$).

5. SEARCH ALGORITHMS

The bicriteria research problem is strongly NP-hard and finding a guaranteed global optimum can require excessive computation time. Since finding the optimal solution with the mathematical method is not always possible, high level meta-search heuristic algorithms can be used to identify optimal/near optimal solution efficiently. However, heuristic algorithms have polynomial time complexity when it comes to large scale problems [77]. In the past decade, meta-heuristics such as tabu search (TS) has been used frequently in literature for solving complex combinatorial problems. These meta-heuristics are flexible and are able to solve large and complex problems [78].

Meta-heuristics can be classified and compared with three basic design choices. First is employing adaptive memory. Adaptive memory algorithms typically solve the problem in a same way as human. Memory-less algorithms do not record the past to compare them with the current state. Tabu search uses adaptive memory. The second feature is the neighborhood exploration. Tabu search uses some systematic neighborhood search either to select the next move or to improve a given solution. The third feature is the number of current solutions carried from one iteration to the next. Population-based strategies process a series of solutions rather than a single solution at each stage. These procedures are classified as evolutionary methods and genetic algorithm (GA) is a good example of these methods. Tabu search moves from one current solution to the next after every iteration. Population-based strategies and adaptive memory strategies are fundamental distinctions that differentiate between heuristics in the literature [79].

5.1. Search Algorithms

Tabu search and genetic algorithm have been used widely for solving scheduling problems. Ruiz and Maroto [40] showed that both tabu search and genetic algorithms are good meta-heuristics for permutation flow-shops, although genetic algorithm needs a good initialization in order to attain a good performance. Vallada et al [80] showed that in permutation flowshop for minimizing tardiness, tabu search and genetic algorithm performed the same. Also they showed that the tabu search methods are good meta-heuristics for this problem. Murata et al [81] showed that for flowshop scheduling with makespan minimization objective function, the genetic algorithm doesn't perform as good as the tabu search. Murata et al [81] and Mantawy et al. [82] also stated that hybrid algorithms outperformed the other algorithms. In hybrid algorithms, the original concepts of heuristic algorithms are modified to include the elements of adaptive memory like in tabu search [79].

In the preliminary experiments conducted for this dissertation, algorithms based on tabu search, genetic algorithm and a hybrid of tabu search-genetic algorithm were considered for developing the search algorithms. The labor constraint introduced significant complexity in defining the characteristics of both tabu search and genetic algorithm. In the first step, a search algorithm was developed based on tabu search (this is the algorithm called TS-TCL in the dissertation). The quality of solutions seemed to be effective, however the search algorithm did not have a good efficiency. In the next step in order to improve the efficiency, a search algorithm based on genetic algorithm was developed. Although the new search algorithm was extremely efficient it showed poor quality compared to the tabu search. In attempts to find an algorithm that is both efficient and effective a hybrid of tabu search-genetic algorithm was pursued. Unfortunately, the hybrid algorithm was less effective and less efficient than the Genetic algorithms. The results comparing these three algorithms are shown in Appendix A.

The results showed that tabu search is a more capable algorithm when it comes to dual resource scheduling and hence more detailed research was performed in order to improve the effectiveness and efficiency of the tabu search. This chapter presents the structure of three different search algorithms based on tabu search in detail.

5.2. Tabu Search Algorithm

The tabu search procedure was proposed by Glover [83- 85], Glover and Kochenberger [86] and Glover and Laguna [87]. Tabu search improves the performance of a local search method by using memory structures. Tabu search has adaptive memory and this is a big plus for it.

Tabu search has been used to solve many scheduling problems in the literature [30-32, 44, 71, 88-89] and all the studies showed that this heuristic is capable of finding the final solution effectively and efficiently. Barnes et al. [90] review applications of TS to various production scheduling problems. TS starts from an initial solution and marches towards better solutions by employing a set of moves for transforming one solution to another. After determining a potential local optimum (solution), it is marked as a tabu and moves cannot be performed on it in the next set of iterations. Tabu search uses neighborhood search procedure to move from one iteration to the other, until the stopping criterion is met. Perhaps the most important type of memory structure used in tabu search is the tabu list. In its simplest form, a tabu list is a short-term memory which contains the solutions or configurations that have been explored in the recent past. As mentioned above the tabu move cannot be performed on the next set of iterations. However, some of these solutions that must now be disregarded because of tabu could have excellent quality and might not have been visited. In order to overcome this issue, the tabu status can be overridden in the event that the value of the selected entry is better than the aspiration level. The aspiration level is the best solution that has been found so far by the search algorithm [77].

5.3. Components of Tabu Search

Initial Solution: an initial solution is required to trigger the search algorithm. The initial solution can be identified arbitrarily or through a systematic mechanism.

Neighborhood Function: This is also called perturbations and it is a change on the seed solution in order to find a neighborhood solution. The neighborhood function identifies a set of new solutions within the neighborhood of a given solution.

Objective Function Evaluation: The goal of the search is to optimize the objective function. Therefore, each time a new solution is generated, its objective function must be evaluated.

Tabu list (TL): After each iteration, the move that resulted in the best solution is entered into the tabu list. The specific move remains as a tabu while it is in the list. In the simplest form the tabu list size (TLS) is 1, which means there is only one entry in the tabu list and it is the last move that was performed in the neighborhood search for identifying the new seed. Without the tabu list, the move will not be considered as a tabu and consequently the search algorithm will cycle back between the parent and child and no improvement can be found. By entering recent moves into the tabu list, the generation of recently visited solutions is avoided. The number of times that a move remains as a tabu depends on the tabu list size. Tabu list is updated by removing the last move recorded before entering the new move as a tabu. As noted before, the tabu list is a short-term memory structure.

Aspiration Level (AL): The aspiration level is the best solution found by the search algorithm so far. As only a part of a solution (neighborhood move) is recorded in the tabu list, it is possible that the search algorithm disregards some good solutions. To prevent this, if the objective function value of the tabu move is better than the aspiration level, the tabu status is overridden and the tabu move is accepted.

Temporary Candidate List (TCL): This is a temporary list that contains the objective function value of neighborhood search. The temporary candidate list is updated after finding a new seed.

Candidate List (CL): The best solution selected among generated neighbors or the best solution among TCL, is recorded in the candidate list. If the move that is selected to enter the CL shows improvement, that solution has the potential of becoming a local optimum. The candidate list is also an explicit memory structure that controls no same solution is recorded in it.

Index List (IL): If a solution becomes a local optimum then it enters the index list. A solution is considered to be a local optimum when its objective function value is at least as good as its parents and at least as good as its child.

Stopping Criteria: The search algorithm terminates when a certain number of local optima have been identified in IL or when the best solution found so far has not improved for a certain number of iterations. In other words the algorithm stops when the number of iterations without improvement (IWI) is more than the maximum number of iterations without improvement (ITM) or the number of entries into the IL is more than the maximum number of entries into the IL (ILM).

The following shows the pseudo code for TS algorithm and TS flowchart is shown in Figure 4.

Step 1. Start
Step 2. Reset CL, IL, TL, IWI, ILM, and ITM
Step 3. Identify the IS
Step 4. Consider the IS as the seed
Step 5. Reset TCL. By using the neighborhood function, find the neighborhood solutions and their objective function values
Step 6. Record the objective function values in the TCL
Step 7. Find the best solution among TCL
Step 8. If the move is tabu go to step 9, otherwise consider the move as a tabu and go to step 10
Step 9. If the solution is worse than AL disregard the solution and find the next best value among TCL and go to step 8; otherwise go to step 10

Step 10. If this solution has already been admitted into the CL, disregard the solution and find the next best value among TCL and go to step 8; otherwise go to step 11.
Step 11. Record the value into the CL
Step 12. If the current CL value is better than its parent, assign a star to the solution and go to step 15; otherwise assign another star to its parent (if the parent already has a star)
Step 13. Record the parent solution with two stars into the IL and record the CL value to the IWI
Step 14. If the stopping criterion is met, find the best solution among IL and stop the search; otherwise go to step 15
Step 15. Consider CL entry as the new seed and go to step 5

5.4. Two Layer Search Algorithm

The meta-heuristic proposed here has two layers, inside and outside layers, which are both based on tabu search. The outside layer or the machine layer disregards the labor resource and attempts to find the best job schedule on machines. The inside layer or labor layer however, tries to find the best schedule of jobs on labor for a specific machine assignment found in the outside layer. As the machine layer is disregarding the labor resource, the solutions found from *only* the machine layer may be infeasible whereas the solution fond from labor layer are always guaranteed to be feasible.

5.4.1. Outside or Machine Layer

As stated before, the machine layer disregards the labor resource and therefore the problem is simplified to a traditional hybrid flowshop (HFS) scheduling problem. However, the traditional HFS problems with sequence-dependent setup times, dynamic release and machine availability times, and stage skipping are among the most complicated type of scheduling problems and are strongly NP-hard. The outside layer is a tabu search procedure that starts with an initial solution and perturbs the solution until a good quality solution is found. The objective function values evaluated in the outside layer do not consider the labor resource and therefore may be infeasible.

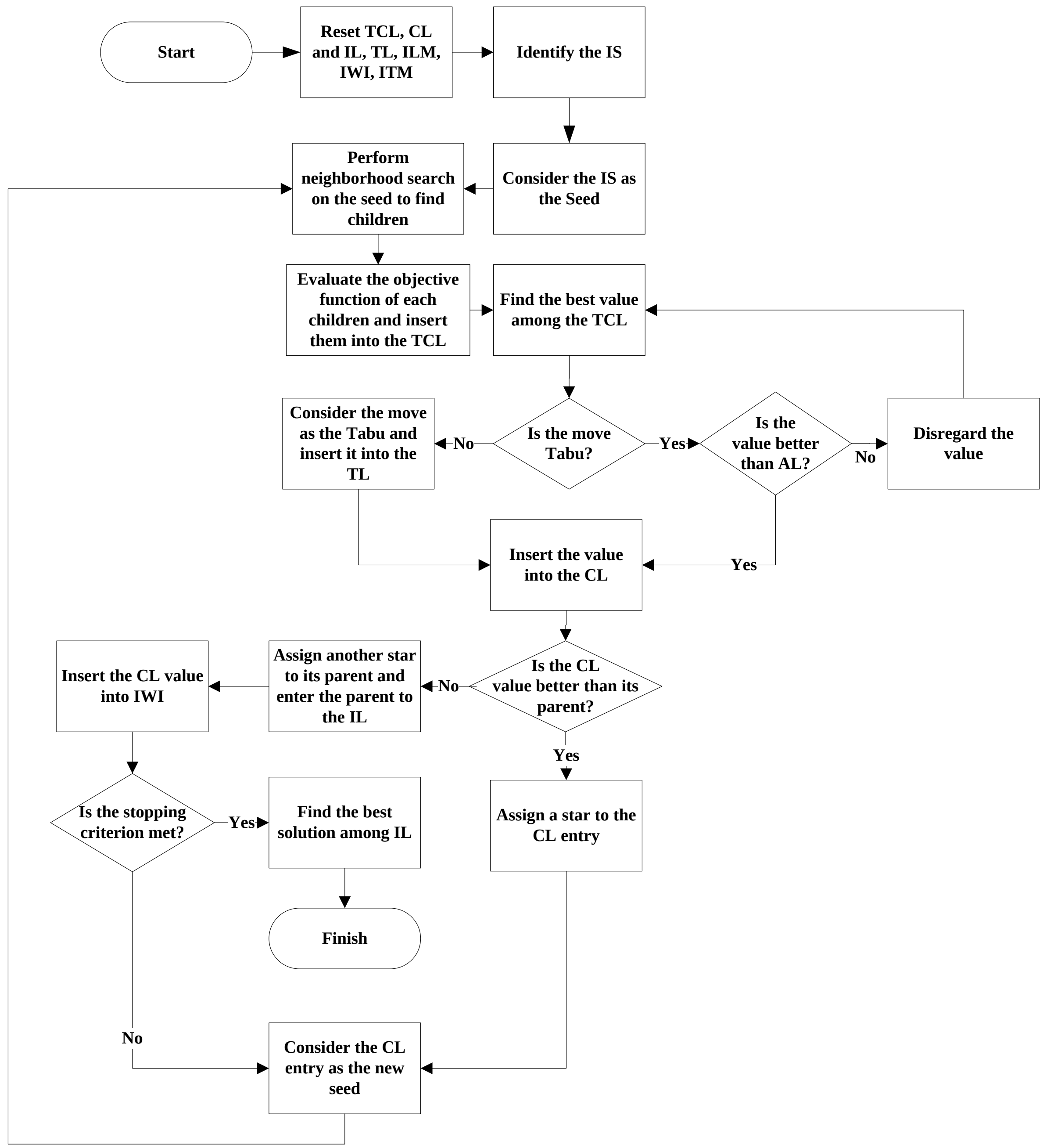

Figure 4. TS algorithm flowchart

5.4.1.1. Initial solution finding mechanism

An initial solution (IS) is needed in order to trigger the search algorithm. Logendran and Subur [30] have shown that the quality of the solution obtained from the search algorithm is dependent on the quality of the IS, so finding a good quality IS is essential. As this research deals with two different goals, finding an IS can be difficult. On the one hand, we are trying to find a schedule that minimizes the total weighted completion time for the producer, and on the other hand, we want to minimize the total weighted tardiness for the customers. For finding IS, the schedule of jobs to the machine is identified for each stage. For the first stage IS has three steps:

- Select the earliest available machine.
- Select the jobs that can be processed on the machine and are released before their setup is finished on the machine. If no such job exists, select a job with smaller release time.
- If more than one job is remaining from the first two steps, the job with the smallest normalized positional value based on the following evaluation is selected; in case of tie, the job with a smaller index is chosen.

For calculating the normalized positional value, two different sequences based on producer and customers' preferences are developed first.

Producer Sequence (PS): the goal of the producer is to minimize the completion time. The shortest processing time rule (SPT) is proven to minimize the total completion time in single machine scheduling problems. The SPT rule is used here to find the producer's sequence. In order to apply this rule, the first step is to change the sequence-dependent setup into sequence-independent setup. To do so, the average setup time of a job on a machine is used as a sequence-independent setup time. The job with the smallest value of $\sum_{t=1}^{T} Average_i(\bar{s}_{ijt} + p_{ijt})\,/w_j$ is scheduled first in this sequence, meaning that the job with the total smallest average processing time with the largest weight is given the highest priority.

Customer Sequence (CS): In this sequence, the earliest due date (EDD) rule is used. EDD is proven to develop the optimal schedule for single machine scheduling problem when the goal is to minimize the maximum tardiness. A weighted EDD rule sequences the jobs based on d_j/w_j. With this rule, the job with the smallest due date and largest weight is scheduled first.

After finding the order of jobs from both PS and CS, the order of jobs in the producer's sequence is multiplied by α and added to the order of jobs in the customers' sequence multiplied by β $(\text{i.e.}, \alpha.PS + \beta.CS)$. The result is a normalized positional value of jobs. Now that the schedule of jobs in the first stage is identified, jobs are sorted by their completion time from the previous stage and the job with a smaller completion time is assigned to the earliest available machine.

5.4.1.2. Neighborhood search

After finding the initial solution, the perturbations are performed on the IS to find neighborhood set and solutions. Perturbations are changes in the structure of the seed (IS). There are four types of moves in machine layer.

Exchange on the same machine: This is simply the exchange of positions of two jobs that are assigned to the same machine. For example, if the sequence of jobs on machine 1 (M1) is J1-J2-J3-J4, then an exchange move of position 2 with 4 will result in J1-J4-J3-J2. Figure 5shows this move.

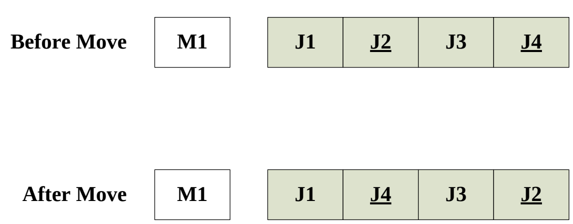

Figure 5. Exchange move on the same machine

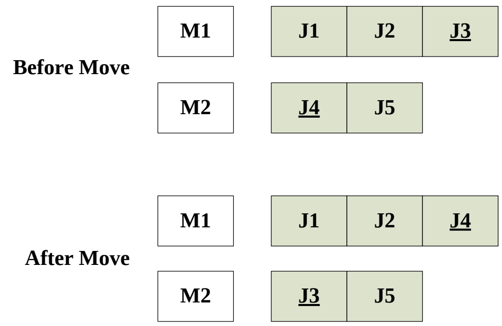

Figure 6. Exchange move on different machines

Exchange on different machines: In this move the positions of jobs are exchanged between two different machines. These kinds of moves are not always feasible. This move is performed if only both machines are capable of performing both jobs. For instance, suppose that the job sequence on M1 is J1-J2-J3 and on M2 is J4-J5. Exchange move of position 3 of M1 with position 1 of M2 is shown in Figure 6.

Insert move on same machine: In this move a position on a machine is inserted into a new position of the same machine. The old and new positions cannot be immediate neighbors or in other words |new position-old position|≥2; otherwise it would be the same as exchange move on the same machine. This also means that this move does not exist if the number of jobs assigned to the machine is less than three. Figure 7shows the insert move of position 1 into position 3 of M1.

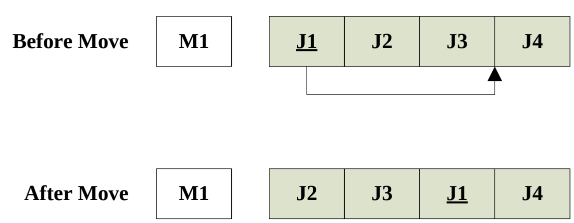

Figure 7. Insert move on the same machine

Insert move on different machines: In this move a position from one machine will be inserted into a new position from another machine. The move is feasible if the new machine is capable of processing the job on the other machine. Figure 8 shows the insert of position 2 from M1 into position 3 of M2.

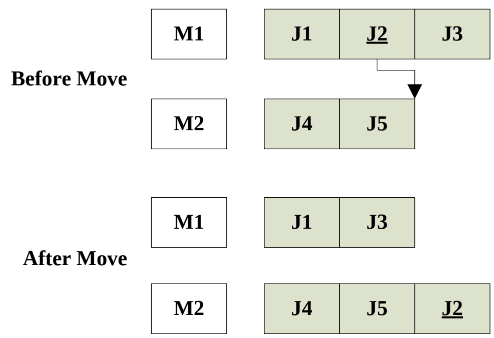

Figure 8. Insert move on different machines

5.4.1.3. Completion time calculation

There are three different cases to consider in completion time evaluation. In case 1, the machine is available after the job is released and so the job completion time is $a_{it} + s_{i0jt} + p_{ijt}$. In case 2, the job is released after the machine is available but its release time occurs prior to the completion of the setup on the machine for that job, so here again the job completion time is calculated with $a_{it} + s_{i0jt} + p_{ijt}$. In case 3, the job is released only after its setup has been completed on the machine, so $r_j + p_{ijt}$ evaluates the job completion time. Or in a simple formula, the completion time is $Max(r_j,\ a_{it} + s_{i0jt}) + p_{ijt}$.

5.4.1.4. Search algorithm

The machine layer starts with IS and after performing the neighborhood search, the objective function value for every move is recorded into the TCL. The best value among the TCL is selected and is inserted into the CL. The selected value cannot enter the CL if the same configuration has already been inserted into the CL. This will lead the search algorithm to produce the same series of solutions that were generated before and will put the search into a loop. The move that resulted in an entry into the CL is considered as a tabu and this move cannot be performed in the next set of iterations. In the event that this solution is better than the AL the tabu status can be overridden. If the value of the current entry into the CL is better than its previous entry (i.e., smaller in objective function value), then the configuration is considered as a potential local optimum and is assigned with a star (*). A potential local optimum will become a local optimum when the next entry into the CL has an objective function value worse than or equal to the potential local optimum, and will be given another star (**). The entries into the CL that receive two stars are local optima and are admitted into the index list (IL). The best solution among the IL is the best final solution for the problem. The search algorithm is terminated when either the number of entries into the IL or the number of iterations without improvement (IWI) reaches the maximum number of entries into the IL (ILM) or the maximum number of IWI (ITM).

5.4.2. Inside or Labor Layer

The labor layer attempts to find the optimal schedule of jobs on labor for a specific machine schedule. When the schedule of jobs on the machine is known, the problem of determining the schedule of jobs on labor in accordance to the machine schedule is also strongly NP-hard. The labor problem in this case is equivalent to an unrelated-parallel machines problem with l unrelated-parallel machines (equivalent to total number of labors) and a maximum of $2*n*T$ jobs (n: total number of jobs, T: total number of stages, 2: setup and run) with sequence-independent setup times, dynamic job release times and a bicriteria objective function, which is strongly NP-hard. For solving this strongly NP-hard problem, a good meta-heuristic search algorithm is needed. Tabu search is used to develop the solutions of labor layer. These solutions are all feasible as they consider both machine and labor resources. The labor layer starts with an internal initial solution (IIS) and tries to find a better quality solution by performing neighborhood search.

5.4.2.1. Internal initial solution finding mechanism

The aim of the inside layer is to find an optimal labor schedule for a specific machine schedule or to evaluate a feasible objective function for the machine schedule. So having a machine schedule is a must. This machine schedule can come from IS, TCL, CL or IL entries. After the machine schedule is known, a feasible schedule of jobs to the labor based on the machine schedule is identified. This initial feasible job schedule on labor is called internal initial solution (IIS). Following procedures shows the steps required for generating IIS:

Step 1. Consider the selected machine schedule
Step 2. Start from stage 1
Step 3. Find the jobs that have the first positions in the stage
Step 4. If there are multiple jobs, choose the job that is assigned to a machine with smaller index
Step 5. Assign both job’s setup and run to their labor
Step 6. If there are more jobs with the same position number, go to step 4; otherwise go to the next position and continue to step 7
Step 7. If there are no jobs in the next position, go to next stage and continue to step 8; otherwise go to step 4
Step 8. If there is no more stages left, exit the procedure; otherwise go to step 3

5.4.2.2. Internal neighborhood search

After finding the internal initial solution the perturbations are performed on the IIS to find neighborhood set and their solutions. Perturbations are changes in the structure of the seed (IIS). There are two types of moves in the labor layer.

Exchange move: This is simply the exchange of positions of two jobs that are assigned to the same labor. For example, if the sequence of jobs on labor 1 (L1) is J1-J2-J3-J4, then an exchange move of position 2 with position 4 will result in J1-J4-J3-J2. Figure 59shows this move.

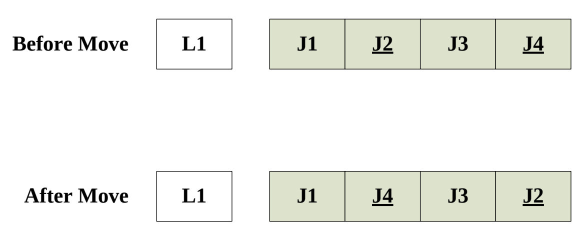

Figure 9. Exchange move on labor

Insert move: In this move a position from one labor will be inserted into a new position of another labor. The move is feasible if the new labor is capable of processing the job. Figure 10 shows the insert of position 2 from L1 into position 3 of L2.

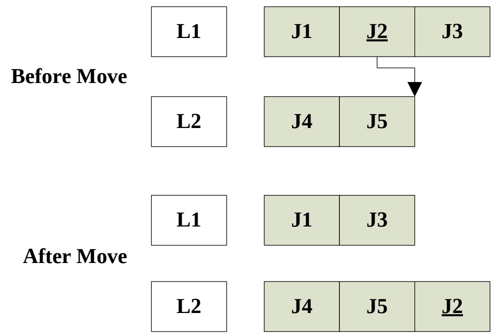

Figure 10. Insert move on labor

5.4.2.3. Completion time calculation

The completion time calculation becomes more complex when dealing with another resource. As the setup and run require different labor skills, it is necessary to evaluate the setup and run completion time separately. The setup start time starts as soon as both machine and labor require for performing the job are available. The setup completion time is $SC_{jt} = Max(a_{it}, L_l) + s_{i0jt}$, where L_l is labor l availability time. The machine and labor availability times will be updated to setup completion time. The run can start as soon as job is released, machine is available, and labor is available. The run completion time is $RC_{jt} = Max(a_{it}, L_l, r_j) + p_{ijt}$. This time, machine, labor availability time and job release time will be updated to run completion time.

5.4.2.4. Search algorithm

The labor layer procedure is very similar to machine layer. It starts with IIS and after performing the neighborhood search, the objective function value for every move is recorded in the ITCL. The best value among the ITCL is selected and is inserted into the ICL. The values of ICL that are as good as their parent and child are considered as a local optimum and are entered into the IIL. The search algorithm is terminated when either the number of entries into the IIL or the number of iterations without improvement (IIWI) reaches the maximum number of entries into the IIL (IILM) or the maximum number of IIWI (IITM), respectively.

The best solution from the IIL can be considered as the feasible solution of the initial fixed machine schedule.

5.5. Hybrid Algorithms

In order to find the solution to the hybrid flowshop scheduling problem with dual resources, both machine and labor layers have to be combined to get a meaningful optimal/near optimal solution. As said before the labor layer can be applied to every single solution found by machine layer (which is TCL) or only to a selected number of machine layer solutions (CL or IL). Each of these applications has their own advantages and disadvantages. In this dissertation three different hybrid algorithms are proposed. In TS-TCL the labor layer is applied to find the feasible solution for every TCL entry, while in TS-CL and TS-IL, labor layer is applied to only the CL and IL entries, respectively.

5.5.1. TS-TCL Algorithm

This algorithm is the most complete and most idealistic way of developing the hybrid algorithm. In this algorithm no single opportunity of finding a better quality solution is neglected. In essence the algorithm is assuming that even a bad quality machine schedule (TCL entries that will not end up being a CL or IL) can have a chance of becoming a good schedule when the labor resource is imposed on it. This idea is motivated by the fact that the optimal schedule of jobs on machines may not necessarily lead to the optimal solution of the whole dual resource problem. As good as this algorithm sounds, there is a big disadvantage to it; the labor layer is a full-fledged complex meta-heuristic based on tabu search, it is faster than the mathematical model but when it is run thousands of times (for every TCL entries) the whole hybrid algorithm becomes very inefficient and slow. As the size of the problem grows, the TS-TCL may not even be able to finish its search within a reasonable time. The pseudo code of the algorithm is as follows:

```
Start
Machine layer
{
        Find IS
        IS → Seed
        Neighborhood search (Exchange and Insert on same and different machines)
        Objective function value → TCL (infeasible)
        Labor Layer
        {
                Find IIS based on TCL machine schedule
                IIS → Internal Seed
                Neighborhood search (Exchange and Insert on Labor)
                Objective function value → ITCL (feasible)
                Best value among ITCL → ICL (feasible)
                Local optima → IIL (feasible)
                Best value among IIL → internal final solution (feasible)
        }
        Internal final solution → TCL (feasible)
        Best value among TCL → CL (feasible)
        Local optima → IL (feasible)
        Best value among IL → Final solution (feasible)
}
End
```

5.5.2. TS-CL Algorithm

This hybrid algorithm attempts to reduce the search time by applying the labor layer only to CL values. The argument in favor of this hybrid algorithm is that there is a little chance that the overall optimal solution is found in bad quality machine schedules. The pseudo code of the algorithm is as follows:

```
Start
Machine layer
{
        Find IS
        IS → Seed
        Neighborhood search (Exchange and Insert on same and different machines)
        Objective function value → TCL (infeasible)
        Best value among TCL → CL (infeasible)
        Labor Layer
        {
```

```
            Find IIS based on CL machine schedule
            IIS → Internal Seed
            Neighborhood search (Exchange and Insert on Labor)
            Objective function value → ITCL (feasible)
            Best value among ITCL → ICL (feasible)
            Local optima → IIL (feasible)
            Best value among IIL → internal final solution (feasible)
        }
        Internal final solution → CL (feasible)
        Local optima → IL (feasible)
        Best value among IL → Final solution (feasible)
}
End
```

5.5.3. TS-IL Algorithm

This algorithm is the most efficient algorithm and is best suited for solving large industry-size problems. The labor layer is applied only on IL entries and therefore the number of times that the labor layer is used in a problem is very limited. The pseudo code of the algorithm is as follows:

```
Start
Machine layer
{
        Find IS
        IS → Seed
        Neighborhood search (Exchange and Insert on same and different machines)
        Objective function value → TCL (infeasible)
        Best value among TCL → CL (infeasible)
        Local optima → IL (infeasible)

        Labor Layer
        {
            Find IIS based on IL machine schedule
            IIS → Internal Seed
            Neighborhood search (Exchange and Insert on Labor)
            Objective function value → ITCL (feasible)
            Best value among ITCL → ICL (feasible)
            Local optima → IIL (feasible)
            Best value among IIL → internal final solution (feasible)
        }
```

Internal final solution → IL (feasible)
Best value among IL → Final solution (feasible)
}
End

5.6. Demonstration of Tabu Search

The steps of the algorithm are demonstrated in a small example with 3 jobs and 3 stages that was introduced in Section 3.1. There are two unrelated-parallel machines in stage 1 (M11 and M21), one machine in stage 2 (M12), and one machine in stage 3 (M13). There are two units of labor, one labor has skill 1 and the other has skill 2.

5.6.1. Outside or Machine Layer

In the outside layer, only the machine-constrained problem is considered and labor is ignored. The goal of the outside layer is to perform a thorough search based on Tabu search on the problem in order to find the best assignment of jobs to machines. Because the labor constraint is ignored in the outside layer, all of the solutions found in the outside layer may be infeasible and cannot be considered as the final solution.

5.6.1.1. Initial solution

For finding the assignment of jobs to machines, first the normalized positional value of jobs is evaluated. For doing this, both the producer sequence (PS) and customer sequence (CS) of jobs are calculated.

As said before, the PS evaluation is based on SPT rule and it is the average weighted processing time for each job. The total weighted processing times are shown in Table 5.1. The order of jobs based on PS is J3, J2, and J1.

Table 5.1 PS evaluation

Average Setup/Run Time on Machines						
	J1		J2		J3	
	Setup	Run	Setup	Run	Setup	Run
M11	21.33	36.00	18.33	41.00	22.33	25.00
M21	26.50	12.00	0.00	0.00	27.50	15.00
M12	13.50	12.00	0.00	0.00	33.00	17.00
M13	7.00	14.00	29.50	30.00	0.00	0.00
Average Setup/Run Time on Stages						
S1	23.92	24.00	18.33	41.00	24.92	20.00
S2	13.50	12.00	0.00	0.00	33.00	17.00
S3	7.00	14.00	29.50	30.00	0.00	0.00
Total	44.42	50.00	47.83	71.00	57.92	37.00
Total Processing Time	94.42		118.83		94.92	
Weight	1		2		2	
Total Weighted Processing Time	**94.42**		**59.42**		**47.46**	

The CS evaluation is based on EDD rule and the weighted due date is used to sort the jobs. Table 5.2 shows the weighted due dates of jobs. The order of jobs based on CS is J3, J2, and J1.

Table 5.2 CS evaluation

Job	J1	J2	J3
Due Date	139	205	109
Weight	1	2	2
Weighted Due date	139	102.5	54.5

The normalized positional value is calculated by $\alpha.PS + \beta.CS$. Table 5.3 shows the calculation for normalized positional value. The final order of jobs is J3, J2, and J1. (α=70% and β=30%)

Table 5.3 Normalized positional values

Job	J1	J2	J3
PS Order	3	2	1
CS Order	3	2	1
Normalized Positional Value	3*0.7+3*0.3=3	2	1

Now that the order of jobs is known, the machine should be selected. The earliest available machine in the first stage is M21 with availability time of 28. M21 is capable of processing J1 and J3. Between J1 and J3, the job that is released before M21 setup time is identified. J1 setup completion time on M21 is 28+16=44 and J1's release time is 15, so J1 has qualifies to be selected. J3's setup completion time on M21 is 28+17=45 and J3's release time is 30, which means J3 is also qualified. Between J1 and J3, J3 is selected because it has smaller normalized positional value. J3 is assigned to M21 with a completion time of 45+15=60. M21's availability time and J3's release time are updated to 60. By using the same concept, J2 is assigned to M11 with a completion time of 39+12+41=92. The last job in stage 1 is J1 and it is assigned to M21 with s completion time of 60+37+12=102. After assigning all jobs to stage 1 and updating the machine availability and job release times, the IS finding mechanism proceeds to stage 2. In stage 2 jobs are sorted based on their completion times from previous stage. Therefore, the order of jobs is J3, J2, and J1. There is only one machine in the second stage and J3 is assigned to M12 first. J3's completion time is 82+36+17=135. J2 is skipping stage 2, so its completion time remains the same as it was in stage 1 at 92. Then J1 is assigned to M12 with a completion time of 135+15+12=162. At the end jobs are assigned to stage 3. The order of jobs is J2, J3, and J1. Again there is only one machine in stage 3 so J2 is assigned to M13 with a completion time of 104+21+30=155. J3 is skipping stage 3 and its completion time will remain the same as it was in stage 2 at 135, and J1's completion time on M13 is 155+12+14=181. Table 5.4 summarizes the calculation of IS objective function value. Figure 11 shows the assignment of jobs to machines.

Table 5.4 IS objective function value evaluation

	Weight	Completion Time	Due date	Tardiness	Weighted completion time	Weighted Tardiness
J1	1	181	139	42	181	42
J2	2	155	205	0	310	0
J3	2	135	109	26	270	52
Normalized Total Weighted Completion Time (70%)						532.7
Normalized Total Weighted Tardiness (30%)						28.2
Initial Solution Objective Function Value						560.9

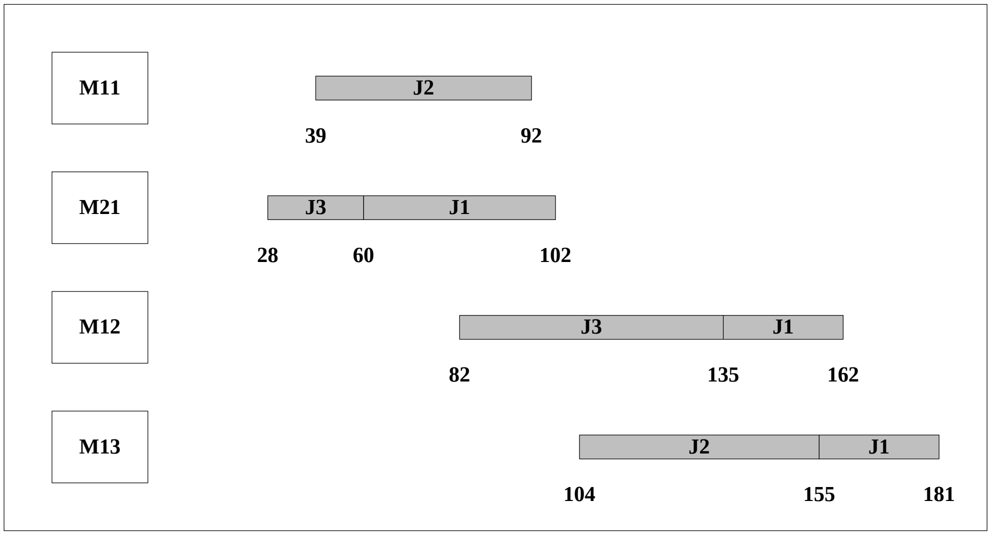

Figure 11. IS schedule of jobs on machines

5.6.1.2. Neighborhood search on machines

The outside or first layer perturbations are performed on the machine. Machine perturbations include exchange on the same machine, exchange between different machines, insert into the same machine and insert into a different machine. The list of all possible perturbations on IS is presented in Table 5.5. There is no exchange move on M11 as J2 is the only job assigned to M11. There is one exchange move between J3 and J1 on M21, one exchange move between J3 and J1 on M12, and one exchange move between J2 and J1 on M13. Exchange of jobs between different machines is not possible in the IS because there is only one machine in stages 2 and 3, and in stage 1, J2 can be processed only by M11 and therefore cannot exchange its place with J3 or J1. Also there are no insert moves on same machines as the maximum number of jobs assigned to each machine is not larger than 2. Insert of jobs on the same machine with less than 3 jobs is the same as exchange move on the same machine and therefore are ignored. There are four insert moves into different machines. Insert moves into different machines are possible only in stage 1, where there is more than one machine in the stage. Insert moves can be performed between M11 and M21. J2 on M11 cannot be inserted into M21 as M21 is not capable of processing J2. However, J1 and J3 on M21 can be inserted into position 1 (before J2) and position 2 (after J2) of M11. After performing each move the new objective function value is calculated and inserted into the TCL. (This is the TCL for machine layer)

Table 5.5 TCL values after the first iteration

Perturbation	TCL	Move	Objective Function
Exchange Same	1	M21: J3 & J1	560.9
	2	M12: J3 & J1	626.9
	3	M13: J2 & J1	703.9
Exchange Different	-	-	-
Insert Same	-	-	-
Insert Different	4	J3 from M21 into the first position of M11	671.3
	5	J3 from M21 into the second position of M11	612.9
	6	J1 from M21 into the first position of M11	570.5
	7	J1 from M21 into the second position of M11	562.9

After finding the TCL, the best value among TCL is considered to be the new seed for the next set of perturbations and will be inserted into the CL. Here the exchange move between J3 and J1 on M21 with objective function value of 560.9 is considered to be the new seed and is inserted into the CL. Also this move is inserted into the tabu list and is tabu in the next couple of iterations. As the tabu list size is only one in this example, this move remains as a tabu only in the next iteration. The new CL objective function value is equal to the previous seed (IS) and has the potential of becoming a local optimum, therefore a star (*) is assigned to this CL entry. Figure 12 shows the configuration of jobs in the new seed (first entry into the CL) and Table 5.6 shows the possible perturbations on the new seed.

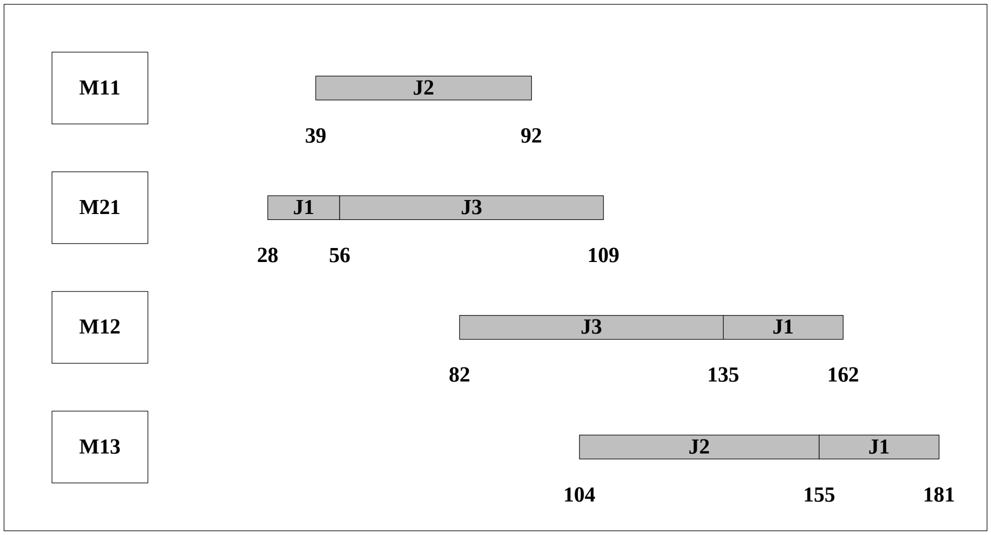

Figure 12. First CL entry's schedule of jobs on machines

Although J3 operation is delayed in stage 1, it does not affect its process in stage 2. The setup of M12 for job 3 is 36 and M12 is available at 82, therefore J3's setup is completed at 118 which is smaller than 109, the release time of J3 from stage 1.

Table 5.6 TCL values after the second iteration

Perturbation	TCL	Move	Objective Function
Exchange Same	**-**	**M21: J1 & J3 –Tabu move**	**Cannot be performed**
	1	M12: J3 & J1	596.9
	2	M13: J2 & J1	703.9
Exchange Different	-	-	-
Insert Same	-	-	-
Insert Different	3	J1 from M21 into the first position of M11	570.5
	4	J1 from M21 into the second position of M11	562.9
	5	J3 from M21 into the first position of M11	671.3
	6	J3 from M21 into the second position of M11	612.9

Exchange of J1 and J3 on M21 is a Tabu move and will not be performed. The best objective function value among the new TCL entries is 562.2, obtained from inserting J1 from M21 into the second position of M11. This configuration is the new seed. The value of the new seed is worse than its parent (562.2>560.9) so another star is assigned to its parent and the parent is inserted into the IL and considered as the first local optimum. The objective function value of the seed is inserted into the CL. The move is entered into the tabu list and is considered as tabu in the next iteration. Table 5.7 shows the CL and IL entries for this example. The algorithm stops after finding 4 local optima.

Table 5.7 CL and IL solutions

ROW	CL	Row	IL	ROW	CL	Row	IL
IS	560.2	IS	560.2	6	587.8**	3	587.8
1	560.9**	1	560.2	7	587.8		
2	562.9			8	589.8		
3	560.9**	2	560.2	9	598.9		
4	570.5			10	596.9**	4	696.9
5	606.5			11	626.9		

5.6.2. Inside or Labor Layer

Now is the time to consider labor in the calculations. The inside layer is a search algorithm based on tabu search that changes the infeasible solutions into feasible solutions by finding the best assignment of jobs to labor. The results in the outside layer do not make any sense without considering the labor in the evaluations. The most complex part of this problem is to consider the labor. The complexity is that the machine and labor schedule cannot be considered simultaneously, so labor schedule is identified after the machine schedule is known. In the outside layer the machine schedule is known in TCL, CL, and IL entries, therefore labor schedule can be considered on TCL, CL, or IL entries. The place (TCL, CL, or IL) that labor schedule is applied, generate different types of TS algorithms. As said before, inside layer finds the best labor schedule. Although, the place that inside layer is used is different in TS algorithms, its application is the same. In TS-TCL, the inside layer is applied to every TCL, in order to find the true objective function value after each machine perturbation. In TS-CL, the inside layer is applied to every CL of the machine constraint problem, and finally in TS-IL, the inside layer is only applied to every IL or to every local optimum found in the machine constraint problem. Clearly, TS-TCL uses the inside layer more frequently than TS-CL and they both use the inside layer more frequently than TS-IL. The procedure is as follows:

- Consider a solution from outside layer. This solution can be TCL, CL, or IL.
- Find an internal initial solution (IIS) which shows the schedule of labor for the fixed machine assignment of the solution in the outside layer.
- Perform the perturbations on the IIL in order to find ITCL, ICL, IIL and the best solution.
- Replace the infeasible solution of the outside layer with the best solution found from the inside layer.

5.6.2.1. Internal initial solution

The first TCL entry from outside layer is chosen as the fixed schedule of jobs to machines. The machine schedule is shown in Figure 12. IIS finding mechanism starts with the first stage. Then the jobs that have first positions are chosen. J1 and J2 are in position 1. From among the multiple jobs, the job that is assigned to the machine with smaller index is selected. J1 is assigned to M21 and J2 is assigned to M11, so J2 is selected. Labor with skill 1 (L1) is required to perform both setup and run of J2 on M11. L1 is available at time zero and therefore J2 setup completion time is 39+12=51 and J2 run completion time is 51+41=92. The next job is J1. J1 also requires L1 for performing both its setup and run. However, L1 is available at 92, so J1's setup completion time is 92+16=108 and J1's run completion time is 108+12=120. Now that all the first positions in stage 1 are assigned to labors, the jobs in the second positions are considered. J3 is in second position on M21. J3 requires L2 for performing its setup and L1 for performing its run. L2 is available at time zero. J3's setup completion time is 120+38=158 and its run completion time is 158+15=173. After all the jobs in different positions of stage 1 are assigned to labor, the IIS mechanism proceeds to the next stage. In stage 2, J3 is in first position. J3 requires L2 for performing its setup and L1 for performing its run. L1 is available at 173 and L2 is available at time 158. J3's setup completion time is

158+36=194 and run completion time is 194+17=211. J1 is in position 2 and it requires L2 for performing its setup and L1 for performing its run. L1 is available at 211 and L2 is available at time 194. J1's setup completion time is 211+15=226 and run completion time is 226+12=238. In stage 3, J2 is in the first position and it requires L2 for performing both its setup and run. L2 is available at 226, so J2's setup completion time is 226+21=247 and run completion time is 247+30=277. J1 has the second position and it needs L1 for both its setup and run. L1 is available at 238 and J1's setup completion time is 277+12=289 and run completion time is 289+14=303. Table 5.8 shows the computation of IIS objective function value. Figure 13 shows the schedule of jobs on both machines and labor. The schedule of jobs only on labor is shown in Figure 14.

Table 5.8 IIS objective function evaluation

	Weight	Completion Time	Due date	Tardiness	Weighted completion time	Weighted Tardiness
J1	1	303	139	164	303	164
J2	2	277	205	72	554	144
J3	2	211	109	102	422	204
Normalized Total Weighted Completion Time (70%)						895.3
Normalized Total Weighted Tardiness (30%)						153.6
Internal Initial Solution Objective Function Value						1048.9

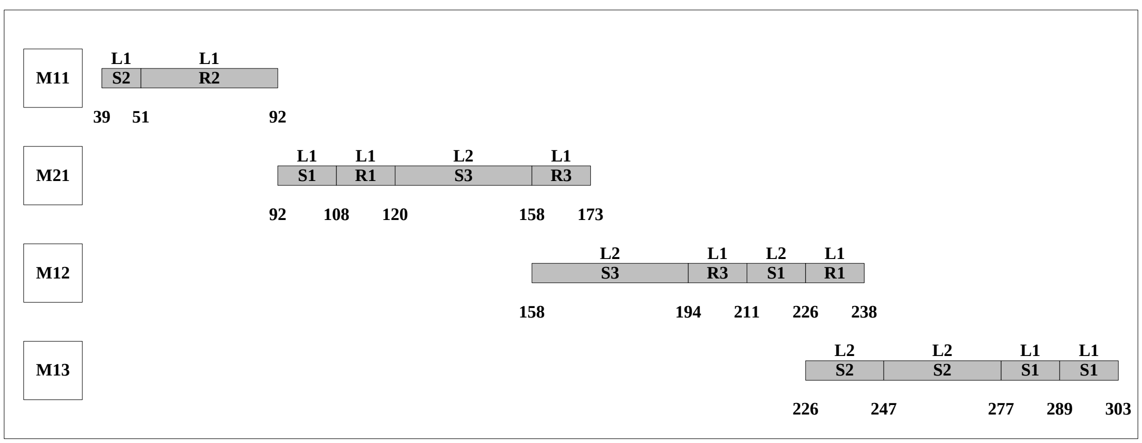

Figure 13. IIS schedule on both machine and labor

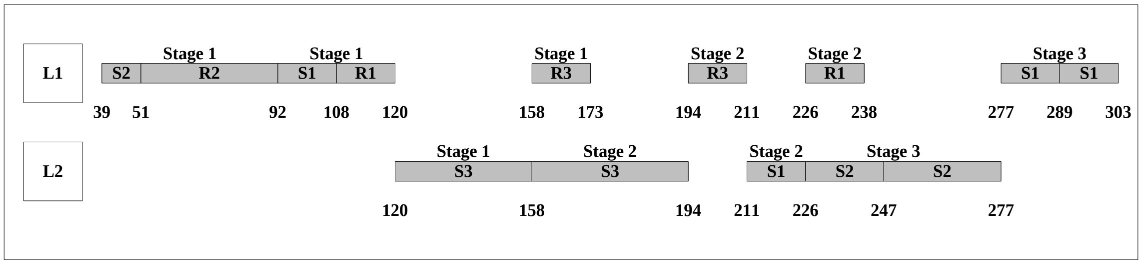

Figure 14. IIS schedule on labor only

5.6.2.2. Neighborhood search on labor

The inside layer perturbations are performed on the labor. Labor perturbations include exchange on the same labor, and insert into a different labor. The list of all perturbations is presented in Table 5.9. For performing labor perturbations it is easier to look at Figure 14. There is no insert move because there is one unit of labor in each skill.

Table 5.9 ITCL objective function values

Perturbation	ITCL	Move	Objective Function
Exchange	1	L1: R2 stage 1 & S1 stage 1	1048.9
	2	L1: R1 stage 2 & S1 stage 3	1060.9
	3	L2: S3 stage 1 & S3 stage 2	943.9
	4	L2: S3 stage 1 & S2 stage 3	880.9
	5	L2: S1 stage 2 & S2 stage 3	997.9
Insert	-	-	-

Exchange of J3's setup in stage 1 with J2's setup in stage 3 has the best objective function value and it is selected to enter the ICL. Also this arrangement is the seed for performing the next iteration. This move is inserted into the internal tabu list and is considered as a tabu in the next labor iteration. Because the objective function of the current seed is better than its parent a star is assigned to this ICL entry to show that it has the potential of becoming a local optimum. Table 5.10 shows the entries into the ICL and IIL. The internal tabu list size is 1 and the internal search algorithm stops after finding 5 local optima.

Table 5.10 ICL and IIL values

Row	ICL	Row	IIL	Row	ICL	Row	IIL
IIS	1048.9	IIS	1048.9	11	662.3**	3	662.3
1	880.9**	1	880.9	12	674.3		
2	880.9			13	683.3		
3	996.3			14	695.3		
4	899.5*			15	740.9		
5	863.9*			16	728.9**	4	728.9
6	829.9*			17	788.9		
7	817.9*			18	737.9*		
8	807.9**	2	807.9	19	727.1**	5	727.1
9	807.9			20	767.1		
10	715.1*						

The Best solution from the inside layer is 662.3. This is the feasible solution for TCL[1] in Table 5.9. The previous TCL value of 560.9 (in the outside layer) is now replaced by 662.3.

5.6.3. Application of Inside Layer in TS-TCL

If the search algorithm is TS-TCL, the inside layer search algorithm is performed for every TCL entry and the infeasible objective function values calculated in the outside layer is replaced with the best solution from inside layer search algorithm. In this case, Table 5.5 is updated to Table 5.11.

Table 5.11 Infeasible vs. feasible TCL values

Row	Infeasible TCL from Outside Layer	Feasible TCL from Inside Layer.
1	560.9	662.3
2	626.9	733.9
3	703.9	805.9
4	671.3	743.9
5	612.9	741.5
6	570.5	683.9
7	562.9	659.5

The outside layer search algorithm continues with the feasible solution for finding CL and IL entries. Table 5.12 shows the updated feasible values of CL and IL entries. The search algorithm stops after 4 entries into the IL.

Table 5.12 Feasible CL and IL values from TS-TCL

Row	CL	Row	IL	Row	CL	Row	IL
IS	621.5	IS	621.5	9	738.9		
1	659.5			10	715.7**	3	715.7
2	621.5**	1	621.5	11	733.9		
3	662.3			12	750.9		
4	683.9			13	754.9		
5	711.3			14	803.9		
6	651**	2	651	15	743.9*		
7	683.5			16	741.5**	4	741.5
8	703.5			17	793.5		

The best solution from TS-TCL algorithm is 621.5 which is also the optimal solution. (The problem was solved with CPLEX [91] and 621.5 was reported as the optimal solution.) TS-TCL took 300 seconds to find 4 entries into the IL.

5.6.4. Application of Inside Layer in TS-CL

If the search algorithm is TS-CL, the inside layer search algorithm is performed on every CL entry and the infeasible objective function values calculated in the outside layer is replaced with the best solution from inside layer search algorithm in CL entries. In this case Table 5.7 is updated to Table 5.13. The outside layer search algorithm continues with the CL feasible solution for finding IL entries.

The best solution from TS-CL algorithm is 621.5 which is also the optimal solution. (The problem was solved with CPLEX and 621.5 was reported as the optimal solution.) TS-CL took 79 seconds to find 4 entries into the IL, which is roughly one-fourth of the time TS-TCL took to find the same solution.

Table 5.13 Feasible CL and IL values from TS-CL

Row	CL	Row	IL	Row	CL	Row	IL
IS	621.5	IS	621.5	9	738.9		
1	662.3			10	703.7		
2	659.5*			11	733.9		
3	621.5**	1	621.5	12	721.5**	3	721.5
4	661.5			13	728.4		
5	711.3			14	766.9		
6	651**	2	651	15	743.9*		
7	683.5			16	741.5**	4	741.5
8	703.5			17	793.5		

5.6.5. Application of Inside Layer in TS-IL

If the search algorithm is TS-IL, the inside layer search algorithm is performed for every IL entry and the infeasible objective function values calculated in the outside layer is replaced with the best solution from inside layer search algorithm in IL entries. In this case Table 5.7 is updated to Table 5.14. The outside layer search algorithm finds feasible IL solution to evaluate the best solution.

Table 5.14 Feasible IL values from TS-IL

	Infeasible IL	Feasible IL
IS	560.2	621.5
1	560.2	662.3
2	560.2	651
3	587.8	703.7
4	696.9	728.4

The best solution from TS-IL algorithm is 621.5 which is also the optimal solution. (The problem was solved with CPLEX and 621.5 was reported as the optimal solution.) TS-IL took 19 seconds to find 4 entries into the IL, which is roughly one-fifteenth of the time TS-TCL took to find the same solution.

5.7. Summary

In this chapter three different hybrid algorithms were developed. Hybrid algorithms have two layers. Both layers are search algorithms based on tabu search. In the outside layer, the schedule of jobs to the machine is identified, and in the inside layer, the schedule of jobs on labor is recognized for a fixed machine schedule. In TS-TCL, the inside layer is applied to TCL entries of the outside layer. In TS-CL, the inside layer is applied to CL entries of the outside layer, and in TS-IL the inside layer is applied to IL entries of the outside layer. We are expecting to see the best quality solutions out of TS-TCL and the worst quality solutions out of TS-IL as in TS-TCL the labor schedule is identified for all machine schedules where as in TS-IL the labor schedule is identified only for local optima. In contrast, TS-IL is supposed to be the most efficient algorithm and TS-TCL to be the least efficient algorithm. In chapter 7 statistical experiments are performed to compare the performance of these algorithms with each other and uncover if there is a statistically significant difference between the effectiveness and efficiency of these algorithms.

6. LOWER BOUNDING METHODS

As stated before, hybrid flowshop scheduling (HFS) problem with dual resources is strongly NP-hard and as the size of the problem grows, number of binary variables increases exponentially. Thus, solving the problem optimally in a polynomial time becomes impossible. In small problems, mathematical solvers such as CPLEX are capable of finding the optimal solution, and the quality of the meta-heuristic algorithms can be compared to the optimal solution. However, in medium to large problems there is no way to assess the quality of meta-heuristics as the optimal solution is not known. In order to evaluate the performance of heuristic algorithms, the solution from the algorithm, which can also be regarded as an upper bound, would need to be compared to a lower bound to assess the quality of the meta-heuristic algorithms. The closer the lower bond is to the optimal solution, the more precise is the assessment of meta-heuristic. Therefore, developing good quality lower bounds is one of the biggest research challenges of this dissertation. Two different lower bounding methods are proposed in this chapter. The first method is logic-based Benders decomposition (LBBD) and the second one is based on iterative selective LP relaxation.

6.1. Logic-Based Benders Decomposition (LBBD)

Logic-based Benders decomposition (LBBD) was introduced by Hooker and Yan [92] in the context of logic circuit verification. The idea was formally developed in [93] and applied to 0-1 programming by Hooker and Ottosson [94]. Jain and Grossmann [95] applied logic-based Benders to scheduling problems in which the subproblems are single machine scheduling problems.

Classical Benders decomposition enumerates values of certain variables for solving the problem. For each set of enumeration, the values of certain variables are fixed and are fed to the subproblem. Solution of the subproblem generates a Benders cut and must be satisfied in all subsequent solutions enumerated. The Benders cut is a linear inequality based on Lagrange multipliers obtained from a solution of the subproblem dual. The process continues until the master problem and subproblem converge in value. [96]

LBBD [94] is a manual decomposition technique that generalizes classical Benders decomposition. The problem is decomposed into a master problem (MP) and a set of subproblems (SPs). The MP is a relaxed version of global problem and the solution from MP is fed into one or more SPs. Each SP generates the tightest lower bound for current MP solution. Solving a problem by LBBD is done iteratively by solving the MP and then solving each SP. If the MP solution satisfies all the bounds generated by the SPs, the MP solution is globally optimal. If not, a Benders cut is added to the MP by at least one violated SP and the MP is resolved.

In classical Benders decomposition [97, 98], the subproblem is always a continuous linear or nonlinear programming problem, and there is a standard way to obtain Benders cuts. In a logic-based Benders method, the subproblem is an arbitrary optimization problem, and a specific scheme for generating cuts must be used for each problem class [99].

Logic-based Benders decomposition applies to problems of the form:

$$\begin{aligned} \text{minimize} \quad & z = f(x, y) \\ \text{subject to} \quad & C(x, y) \\ & x \in D_x, y \in D_y \end{aligned} \tag{1}$$

Where $C(x, y)$ is a set of constraints containing variables *x*, *y*. D_x and D_y denote the domains of *x* and *y*, respectively. When *x* is fixed to a given value $\bar{x} \in D_x$, the following subproblem results:

$$\begin{aligned} \text{minimize} \quad & v = f(\bar{x}, y) \\ \text{subject to} \quad & C(\bar{x}, y) \\ & y \in D_y \end{aligned} \tag{2}$$

Here, $C(\bar{x}, y)$ is the constraint that results from fixing $x = \bar{x}$ in $C(\bar{x}, y)$.

The solution of subproblem (2) is the tightest possible lower bound ($\hat{v}$) on $f(\bar{x}, y)$ from $C(\bar{x}, y)$. The basic idea of Benders decomposition is to derive a bound $B_{\bar{x}}(x)$ for other values of *x*. In classical Benders, the same linear combination is used. In general, the bounding function $B_{\bar{x}}(x)$ should have two properties:

Property 1: $B_{\bar{x}}(x)$ provides a valid lower bound on $f(x, y)$ for any given $x \in D_x$. That is, $f(x, y) \geq B_{\bar{x}}(x)$ for any feasible (x, y) in (1).

Property 2: In particular, $B_{\bar{x}}(\bar{x}) = \hat{v}$.

If *z* is the objective function value of (1), the valid inequality $z \geq B_{\bar{x}}(x)$ is a Benders cut. In iteration *H* of the Benders algorithm, a master problem is solved whose constraints are the Benders cuts generated so far:

$$\begin{aligned} \min \quad & z \\ \text{subject to} \quad & z \geq B_{x^h}(x), \;\; h = 1, \ldots, H-1. \\ & z \in R, \;\; x \in D_x \qquad (3) \end{aligned}$$

Here, $x^1, \ldots, x^{H-1}$ are the solutions of the previous $H-1$ master problems. Then, the solution $\bar{x}$ of (3) defines the next subproblem (2). [100]

6.1.1. Application of LBBD to HFS

The HFS with dual resources can be decomposed into machine-master problem and labor subproblem. In the machine-master problem, the schedule of jobs to machines is specified. This schedule results in labor subproblem where the schedule of jobs on labor is identified based on the master problem machine schedule. A mixed-integer linear programming (MILP) model will be presented for the master problem. The use of MIP takes advantage of operations research tools (such as CPLEX) to obtain tight lower bounds for the HFS with dual resources. Another MILP model is used to solve the subproblem. The subploblem generates an upper bound for the global problem and a lower bound for the master problem solution. If the MP and SP solutions are equal the optimal solution is found; otherwise a benders cut is obtained from SP. Figure 15 shows the process of LBBD for the HFS problem.

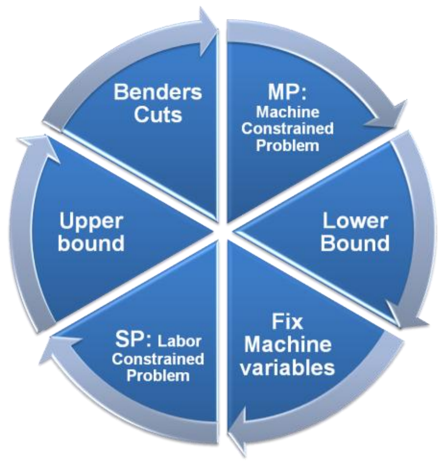

Figure 15. LBBD process for the HFS problem

6.1.2. Machine Master Problem

The MILP formulation of the master problem is a relaxation of the HFS with dual resources. In the master problem, the labor resource is relaxed and the master problem is a traditional HFS with machine as the only resource. This relaxed model identifies schedule of jobs on machines. The master problem is:

$min \quad z = \alpha \sum_{j=1}^{n} w_j RC_{jT} + \beta \sum_{j=1}^{n} w_j TD_j$ (4)

$\sum_{i=1}^{m_t} x_{ijt} = o_{jt}, \; j = 1,2,\ldots,n \;\; t = 1,2,\ldots,T.$ (5)

$SB_{jt} \geq a_{it} x_{ijt}, \quad i = 1,2,\ldots,m_t. \; j = 1,2,\ldots,n \;\; t = 1,2,\ldots,T.$ (6)

$SC_{jt} \geq SB_{jt} + s_{i0jt} y_{i0jt}, \quad i = 1,2,\ldots,m_t. \; j = 1,2,\ldots,n \;\; t = 1,2,\ldots,T.$ (7)

$SC_{jt} - SB_{jt} + M(1 - y_{ikjt}) \geq s_{ikjt} x_{ijt}, \; i = 1,\ldots,m_t. \; j,k = 1,\ldots,n. k < j \;\; t =$
$1,2,\ldots,T.$ (8)

$SC_{kt} - SB_{kt} + M(1 - y_{ijkt}) \geq s_{ijkt}x_{ikt},\ i = 1, \dots, m_t.\ j, k = 1, \dots, n. k < j\ \ t = 1,2, \dots, T.$ (9)

$SB_{jt} - RC_{kt} + M\left(1 - y_{ikjt}\right) \geq 0,\ i = 1, \dots, m_t.\ j, k = 1, \dots, n. k < j\ \ \ t = 1,2, \dots, T.$ (10)

$SB_{kt} - RC_{jt} + M\left(1 - y_{ijkt}\right) \geq 0,\ i = 1, \dots, m_t.\ j, k = 1, \dots, n. k < j\ \ t = 1,2, \dots, T.$ (11)

$\sum_{j=1,k\neq j}^{n} y_{ikjt} \leq 1,\ \ i = 1,2, \dots, m_t,\ \ k = 0,1,2, \dots, n\ \ t = 1,2, \dots, T.$ (12)

$\sum_{k=0,k\neq j}^{n} y_{ikjt} = x_{ijt},\ \ i = 1,2, \dots, m_t,\ \ j = 1,2, \dots, n\ \ t = 1,2, \dots, T.$ (13)

$y_{ikjt} \leq x_{ikt},\ \ \ \ \ i = 1,2, \dots, m_t,\ \ j, k = 1,2, \dots, n, k \neq j\ \ t = 1,2, \dots, T.$ (14)

$RB_{jt} \geq SC_{jt},\ \ j = 1,2, \dots, n\ \ t = 1,2, \dots, T.$ (15)

$RB_{jt} \geq r_j x_{ijt},\ \ i = 1,2, \dots, m_t.\ \ \ j = 1,2, \dots, n\ \ t = 1,2, \dots, T.$ (16)

$RC_{jt} \geq RB_{jt} + p_{ijt}x_{ijt},\ \ \ \ i = 1,2, \dots, m_t.\ \ j = 1,2, \dots, n\ \ t = 1,2, \dots, T.$ (17)

$RB_{jt} \geq RC_{jt-1}\ ,\ \ j = 1,2, \dots, n\ \ t = 2,3, \dots, T.$ (18)

$TD_j \geq RC_{jT} - d_j,\ \ \ \ \ \ j = 1,2, \dots, n.$ (19)

The above constraints (5-19) come directly from machines constraint introduced in chapter 4 MIP model for the HFS problem. In order for the MP search to be more than just a blind enumeration of its solution space, some relaxation of the SP (labor model) should be present in the MP model. In order to develop a relaxed version of subproblem a closer attention should be paid to the subproblem. The subproblem is only responsible for identifying the labor schedule, the labor does not impose any setup or run time on the job (they are all inherited from the master (machine) problem). Job completion times increase because of labor conflict. A unit of labor cannot process two different jobs at the same time. In other words, if in machine schedule, the processing of two jobs which need the same labor has overlap, one job has to wait until the other job is finished with its processing, and then the other job can start its processing. In this model the objective function is minimization of both total weighted completion time and total weighted tardiness. In order to evaluate the objective function, the completion time of each job should be known. Finding a relaxed version of the labor subproblem to include in MP that is capable of finding an estimate of each job's completion time is as complex as solving the whole labor subproblem. However, an estimate of minimum flow time on labor schedule is fairly straight forward. When assignment of job to machines is known, the assignment of job to labor skill is also known. (When a job is assigned to a machine, there is only one labor skill capable of performing the job; however if the job assignment to a labor skill is known, there can be multiple machines capable of processing that job.) Also because the schedule of jobs on machines is identified the exact setup and run time of each job is known. In addition, a unit of labor cannot process two jobs at the same time and in the best case scenario there is no delay between processing of two jobs on a labor. The minimum estimated flow time on labor is the sum of setup or run times of jobs assigned to that labor. (Note that setup and run of a job is not necessarily assigned to the same labor.) Overall, the real flow time of the problem is never smaller than the minimum estimated flow time on labor. If the flow time obtained from the machine master problem is smaller than the minimum estimated labor flow time, it means that the completion times are not large enough and the lower bound found from the machine master problem can be improved. When labor flow time is larger than machine flow time, there is no exact information on comparison between different job completion times on labor and machine, but one thing is clear: the largest completion time on machine is smaller than or equal to the largest completion time on labor. However, these jobs can be completely different jobs. Because job index is not known, no specific weight or due date can be assigned to the largest completion time on labor. The objective function is a weighted objective function, evaluated as the weighted sum of weighted completion times and weighted tardiness. At least the difference between the labor and machine flow times can be added to the producer part of objective function, but what weight should be used? In the worst-case scenario, the difference between the flow times should come from jobs with the smallest weight. For the customer part of the objective function, the tardiness should be evaluated. In the worst case the tardiness is the difference between the flow time and the largest due date, so the value that can be added to the objective function is the difference between maximum tardiness on machine part and the estimated tardiness from labor. Minimum weight is assigned to tardiness, following the same concept as the completion time. The following model shows the revised master problem, formulated as a result of adding the relaxed version of the subproblem model to the original master problem. In other words, the revised master problem, with (20) as its objective function instead of (4) and (5) – (19) and (21) – (28) as its constraints, is indeed the one that establishes the link between the machine master problem and the labor subproblem formulated below.

$$min \quad z = \alpha(\sum_{j=1}^{n} w_j RC_{jT} + min_j\, w_j\, VC) + \beta(\sum_{j=1}^{n} w_j TD_j + min_j\, w_j\, VT) \quad (20)$$

$$F \geq \underset{l}{\mathrm{Max}}\{\min_j r_j + \sum_{t=1}^{T} \sum_{j=1}^{n} \sum_{i=1}^{m_t} \left[g_{ijlt} \sum_{k=0}^{n} s_{ikjt} y_{ikjt} + g'_{ijlt} p_{ijt} x_{ijt}\right]\}, \quad (21)$$

$$MC \leq RC_{jT} + M\left(1 - e_j\right), \quad j = 1,2, \dots, n. \quad (22)$$

$$\sum_{j=1}^{n} e_j = 1, \quad (23)$$

$$VC \geq F - MC, \quad (24)$$

$$MT \leq TD_j + M\left(1 - b_j\right), \quad j = 1,2, \dots, n. \quad (25)$$

$$\sum_{j=1}^{n} b_j = 1, \quad (26)$$

$$VT \geq F - \max_j d_j - MT, \quad (27)$$

Cuts (28)

Where,

F is the minimum estimated labor flow time

MC maximum completion time on machines or flow time from machine part

e_j a binary variable. 1 if job j has the maximum completion time, and 0 otherwise

VC difference between labor and machine flow times

MT maximum tardiness on machines

b_j a binary variable. 1 if job j has the maximum tardiness, and 0 otherwise

VT difference between labor and machine tardiness for the job with the largest tardiness

Constraint (20) evaluates the updated objective function. It is the weighted sum of weighted completion times and weighted tardiness. Also the difference between labor and machine flow times is added to the producer part and the difference between labor and machine tardiness is added to the customer part. The weight assigned to these new components of objective function is the minimum weight. Constraint (21) estimates the minimum flow time on labor. The labor flow time is the maximum of summation of setup and run time assigned to each labor. No job can start its process sooner than its release time and that is why the minimum release time of jobs is added to all labor flow times. g_{ijlt} is a parameter and shows that labor *l* is capable of processing job *j* setup on machine *i* at stage *t*. y_{ikjt} is a variable that shows job *j* is processed immediately before job *k* on machine *i* at stage *t*. s_{ikjt} is the setup time of job *j* that is processed after job *k* on machine *i* at stage *t*. The whole term of $\sum_{t=1}^{T}\sum_{j=1}^{n}\sum_{i=1}^{m_t} g_{ijlt}\sum_{k=0}^{n} s_{ikjt}y_{ikjt}$ simply adds up setup times if they are assigned to labor *l*. The same argument works for run times. Constraints (22) and (232) jointly find the maximum completion time and constraint (24) calculates the difference between machine and labor flow times. Likewise, constraints (25) and (26) jointly find the maximum tardiness and constraint (27) calculates the difference between maximum machine and labor tardiness.

6.1.3. Labor Subproblem

Once a solution to the master problem is found, the schedule of jobs on machines is known.

An MILP model is used to formulate the subproblem. The model will create the schedule of jobs on labor such that the bicriteria objective function is minimized. The labor model is equivalent to unrelated-parallel machine scheduling with *l* machines (equivalent to number of labor) and *2*n*T* jobs (*2*: setup and run, *n*: number of jobs, and *T*: number of stages) with sequence-independent setup time and dynamic release times. The formulation of the subproblem is shown below.

min $W = \alpha \sum_{j=1}^{n} w_j RC_{jT} + \beta \sum_{j=1}^{n} w_j TD_j$ (29)

$\sum_{l=1}^{p} u_{jlt} = o_{jt}, \; j = 1,2,\dots,n' \quad t = 1,2,\dots,T'$. (30)

$\sum_{l=1}^{p} v_{jlt} = o_{jt}, \; j = 1,2,\dots,n' \quad t = 1,2,\dots,T'$. (31)

$u_{jlt} \le \sum_{i=1}^{m_t} g_{ijlt}.x_{ijt}, \; l = 1,2,..,p, j = 1,2,\dots,n', t = 1,2,\dots,T'$. (32)

$v_{jlt} \le \sum_{i=1}^{m_t} g'_{ijlt}.x_{ijt}, \; l = 1,2,..,p, j = 1,2,\dots,n', t = 1,2,\dots,T'$. (33)

$SB_{jt'} - SC_{kt} + M(1 - z_{ktjt'l}) + M(2 - u_{jlt'} - u_{klt}) \ge 0,$ (34)

$SB_{kt} - SC_{jt'} + Mz_{ktjt'l} + M(2 - u_{jlt'} - u_{klt}) \ge 0,$ (35)

$RB_{jt'} - SC_{kt} + M(1 - z'_{ktjt'l}) + M(2 - v_{jlt'} - u_{klt}) \ge 0,$ (36)

$SB_{kt} - RC_{jt'} + Mz'_{ktjt'l} + M(2 - v_{jlt'} - u_{klt}) \ge 0,$(37)

$SB_{jt'} - RC_{kt} + M(1 - q_{ktjt'l}) + M(2 - u_{jlt'} - v_{klt}) \ge 0,$ (38)

$RB_{kt} - SC_{jt'} + Mq_{ktjt'l} + M(2 - u_{jlt'} - v_{klt}) \ge 0,$ (39)

$RB_{jt'} - RC_{kt} + M(1 - q'_{ktjt'l}) + M(2 - v_{jlt'} - v_{klt}) \ge 0,$ (40)

$RB_{kt} - RC_{jt'} + Mq'_{ktjt'l} + M(2 - v_{jlt'} - v_{klt}) \ge 0,$ (41)

$l = 1,\dots,p, j,k = 1,\dots,n', t,t' = 1,2,\dots,T', (k < j \; for \; \forall t,t') \cup (k = j \; for \; t < t')$.

$SB_{jt} \ge a_{it}x_{ijt}, \quad i = 1,2,\dots,m'_{t'}. \; j = 1,2,\dots,n' \; t = 1,2,\dots,T'$. (42)

$SC_{jt} \ge SB_{jt} + s_{i0jt}y_{i0jt}, \quad i = 1,2,\dots,m'_{t'}. \; j = 1,2,\dots,n' \; t = 1,2,\dots,T'$. (43)

$SC_{jt} - SB_{jt} + M\left(1 - y_{ikjt}\right) \ge s_{ikjt}x_{ijt}, \; i = 1,\dots,m'_{t'}. j,k = 1,\dots,n'. k < j, t = 1,2,\dots,T'$. (44)

$SC_{kt} - SB_{kt} + M(1 - y_{ijkt}) \ge s_{ijkt}x_{ikt}, i = 1,\dots,m'_{t'}. j,k = 1,\dots,n'. k < j, t = 1,2,\dots,T'$. (445)

$SB_{jt} - RC_{kt} + M\left(1 - y_{ikjt}\right) \ge 0, \; i = 1,\dots,m'_{t'}. j,k = 1,\dots,n'. k < j, t = 1,2,\dots,T'$. (46)

$SB_{kt} - RC_{jt} + M\left(1 - y_{ijkt}\right) \ge 0, \; i = 1,\dots,m'_{t'}. j,k = 1,\dots,n'. k < j, t = 1,2,\dots,T'$. (47)

$RB_{jt} \ge SC_{jt}, \; j = 1,2,\dots,n' \; t = 1,2,\dots,T'$. (48)

$RB_{jt} \ge r_j x_{ijt}, \; i = 1,2,\dots,m'_{t'}. \; j = 1,2,\dots,n' \; t = 1,2,\dots,T'$. (49)

$RC_{jt} \ge RB_{jt} + p_{ijt}x_{ijt}, \quad i = 1,2,\dots,m'_{t'}. \; j = 1,2,\dots,n' \; t = 1,2,\dots,T'$. (50)

$RB_{jt} \geq RC_{jt-1}$, $j = 1,2, \dots, n', \ t = 2,3, \dots, T'$. (51)

$TD_j \geq RC_{jT} - d_j, \qquad j = 1,2, \dots, n'$. (52)

Where x_{ijt} and y_{ikjt} are parameter values s obtained from the master problem. The set of jobs to schedule is n' on $m'_{t'}$ machines at T' stage, which is the schedule chosen in the master problem.

Constraint (29) shows the objective function of the subproblem. Constraints (30)-(41) are the MILP labor constraints from the MILP model in Chapter 4, and constraints (42)-(52) are additional linear constraints inherited from the master problem.

The subproblem generates a lower bound for a machine schedule obtained from the master problem and an upper bound for the global problem.

6.1.4. Cuts

If the objective function (*W*) found in the subproblem is equal to the master problem's objective function (*z*), then this is the optimal solution and the procedure stops. In the case where the *W* is greater than *z*, a cut is created and sent to the master problem. The master problem is then re-solved with the added cut. The cut from such a subproblem in an iteration *h* is:

$$z \geq W_h^* \left\{1 - \left(\sum_{i \in m'_{t'}} \sum_{j \in n'} \sum_{t \in T'} \left(1 - x_{ijt}\right) + \sum_{i \in m'_{t'}} \sum_{k \in n'} \sum_{j \in n'} \sum_{t \in T'} \left(1 - y_{ikjt}\right)\right)\right\},$$

$h = 1, \dots, H - 1$ (53)

Here, z is the objective function of the master problem and W_h^* is the objective function found in iteration h when solving the subproblem. The cut states that the future solutions of the master problem can only decrease the objective function if another schedule of jobs to machines is given. That is, if the same assignment is given to the subproblem, the x_{ijt} and y_{ikjt}variables that are part of this cut will all be equal to 1. If this is the case, then $(1 - x_{ijt})$ = 0 and $(1 - y_{ikjt})$ = 0 for all i, j, and t and the objective function of the subproblem becomes a lower bound on z. When a different schedule is made and at least one of the x_{ijt} or y_{ikjt} variables that previously had a value of 1 is 0, the cut becomes redundant as the right hand side will be at most zero.

This cut follows the two conditions defined by Chu and Xia [101] to be a valid cut; the cut removes the current solution from the master problem and does not eliminate any global optimal solutions.

The cut presented is a type of no-good cut [102] stating that the current solution is not optimal and so is removed from the search space.

6.2. Iterative Selective LP Relaxation

As the problem becomes larger, the MILP solver cannot provide an optimal solution within the time limit of eight hours. Besides, the gap between the upper bound and lower bound reported at the terminating point is usually high. Alternative methods for finding the optimal solutions or lower bounds should be applied. Logic-based Benders decomposition is used to find an optimal solution or a good lower bound in this problem. However, the HFS problem with dual resources is very complex and contains a large number of binary and general variables. Even with LBBD, it still requires a very long computation time in some large problems, although the lower bound found from LBBD is better than the lower bound from MILP solvers. The LP relaxation of a MILP problem can provide a lower bound within a short time. However, the lower bound obtained is typically very poor in quality. Therefore, a selective LP relaxation (LPR) is used instead. In this method, some of the binary variables such as x_{ijt}, y_{ikjt}, u_{jtl}, v_{jtl}, $z_{ktjt'l}$, $z'_{ktjt'l}$, $q_{ktjt'l}$, or $q'_{ktjt'l}$ are LP relaxed. That is, some of the binary variables are allowed to be fractional in the bounds of 0 and 1.

The iterative selective LP relaxation method is an extended version of the selective relaxation method. After a solution is found from the complete LP relaxed problem, a selected group of the positive/fractional relaxed variables (i.e., potential variables) in the solution are restricted to be binary again. Therefore, the number of binary variables in the relaxed problem is increased in every iteration. The lower bound obtained is therefore tighter than that from the selective relaxation method. This process is repeated until two consecutive equal solutions are found or the limit on the total computation time of eight hours is reached. Figure 16 shows the procedure of the iterative selective LP relaxation.

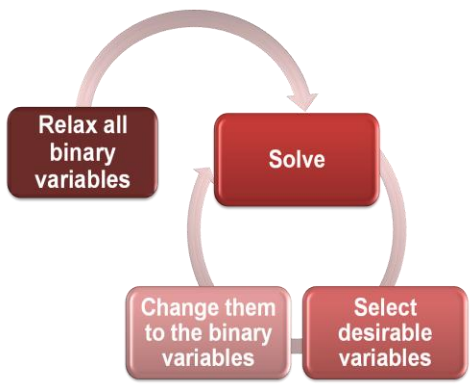

Figure 16. Iterative selective LP relaxation

If in each iteration only a selected group of *fractional* relaxed variables are changed to binaries, the method is called fractional LP relaxation (F-LPR) and if only a selected group of *positive* relaxed variables are changed to binaries, the method is called positive LP relaxation (P-LPR). The decision about how to select these groups of variables is described below.

In this research binary variables are divided into two categories: assignment and sequence variables. The assignment variables identify the assignment of jobs to machines with x_{ijt} or to labor with both u_{jtl} and v_{jtl}. The sequence variables show the sequence of jobs on machines with y_{ikjt} or on labor with $z_{ktjt'l}$, $z'_{ktjt'l}$, $q_{ktjt'l}$, and $q'_{ktjt'l}$. The selection decision is made based on sequence variables only. After solving an LP relaxed problem at any iteration, if a relaxed *sequence* variable is positive/fractional it will be restricted to a binary. However, if a relaxed sequence variable is 0 (in positive LP relaxation) or 0 and

1 (in fractional LP relaxation) its correspondent relaxed positive/fractional *assignment* variables will be restricted to binaries. The reason why decisions are based on sequence variables only is that when two jobs are assigned to a resource, (ex. Job 1 and 2 are assigned to machine 1 at stage 3, $x_{113} = 1, x_{123} = 1$) they might not necessarily be processed one after the other ($y_{1123} = 0/1\ or\ y_{1213} = 0/1$). However, if two jobs are sequenced one after the other on a resource (Job 1 is processed before job 2 on machine 1 at stage 3, $y_{1123} = 1$), they are both assigned to that resource too ($x_{113} = 1, x_{123} = 1$). In other words, by knowing the sequence variable, assignment variables can be identified, however, by knowing the assignment variable, the sequence variable cannot be guessed.

6.2.1. Demonstration of Fractional LP Relaxation

This section describes the step by step application of fractional LP relaxation to the small example problem introduced in section 5.5. In the first step, the LP mathematical model is developed by using the MILP model in Chapter 4. Then all of the binary variables are relaxed in bounds of 0 and 1. The proposed LP problem was solved by CPLEX and the objective function value of 408.8 was obtained as the first lower bound. (The optimal solution of the MILP model is 622.5) In the next step, decision about either restricting a variable to a binary or keeping it as a real variable for the next iteration should be made. Table 6.1 shows the decisions on machine variables.

After looking at the sequence variables and deciding about restricting either assignment or sequence variables to binary, it is possible that some assignment variables remain untouched as required in the formulation of the original problem. These undecided variables will keep their type. In this iteration $x_{112}, x_{113}, x_{123}, and\ x_{132}$ remain and they all are real variables and they will continue to remain real in the next iteration. Interestingly, the value of all of them is equal to 1, but even if they were fractional we would not have changed them to binaries. In row 1, $y_{1011} = 0$, and it is not a fractional value, so it will be kept as a real variable. Its correspondent assignment variable is $x_{111} = 0.25$ (this is the first job so there is only one correspondent assignment variable).

The assignment variable is fractional so it is restricted to a binary. In row 3, $y_{1021} = 0.75$ has a fractional value and it is restricted to a binary, the procedure does not change its correspondent assignment variables, because the hope is that in the next iteration, the assignment variables will have a 0 or 1 value because their sequence variable is restricted to a binary. In row 18 $y_{1321} = 0$, it is not fractional and will stay real, its assignment variables are $x_{121} = 1\ and\ x_{131} = 0.25$, x_{121} is not fractional so it will stay as a real variable but x_{131} is fractional and is restricted to a binary.

Table 6.1 Selective LP relaxation from iteration 1 to 2 on machine variables

Row	Sequence variables	Value	Current iteration	Next iteration	Assignment variables	Value	Current iteration	Next iteration
1	y1011	0	real	Real	x111	0.25	real	binary
2	y2011	0	real	Real	x211	0.75	real	binary
3	y1021	0.75	real	binary				
4	y1031	0	real	Real	x131	0.25	real	binary
5	y2031	0	real	Real	x231	0.75	real	binary
6	y1012	0.00017	real	binary				
7	y1032	0.0003	real	binary				
8	y1013	0.0003	real	binary				
9	y1023	0.00038	real	binary				
10	y1121	0.25	real	binary				
11	y1131	0	real	Real	x111 x131	0.25 0.25	real real	binary* binary*
12	y1231	0.25	real	binary				
13	y2131	0.75	real	binary				
14	y1132	0.9997	real	binary				
15	y1123	0.99962	real	binary				
16	y1211	0.25	real	binary				
17	y1311	0	real	Real	x111 x131	0.25 0.25	real real	binary* binary*
18	y1321	0	real	Real	x121 x131	1 0.25	real real	real binary*
19	y2311	0.75	real	binary				
20	y1312	0.99983	real	binary				
21	y1213	0.9997	real	binary				

* Means the assignment variable has been already forced to be a binary by another sequence variable.

The process is the same on the labor variables. After decisions are made about either to keep a variable as real or to restrict it to a binary, the new MILP relaxed model is solved by CPLEX. In this example the procedure is continued for 5 iterations, and it stops after 0.5 second because the objective function values at iteration 4 and 5 are the same. The lower bound found from this method is 621.5 which is also the optimal value. Table 6.2 shows the progress of lower bound in different iterations of fractional LP relaxation.

Table 6.2 Lower bounds found from fractional LP relaxation at every iteration

Iteration	Lower bound
1	408.8
2	575.9
3	609.5
4	621.5
5	621.5

6.3. Summary

In this chapter two different lower bounding methods, one based on logic-based Benders decomposition and one based on iterative selective LP relaxation, are developed. LBBD is capable of identifying the optimal solution. However, this method can be slow in some of the problems as the structure of master problem and subproblem are still strongly NP-hard. This means that finding an optimal solution is not always possible and the method needs to be stopped prematurely. Although the optimal solution cannot be found, the quality of the lower bound obtained from this method is usually better than the lower bound found from MILP solver. The iterative selective LP relaxation method can only obtain a lower bound and it is very fast in finding a good lower bound but the quality of the lower bound found from this method can be a little inferior to LBBD.

7. COMPUTATIONAL EXPERIMENTS

In this chapter, computational experiments are performed in order to determine the best algorithm among the three proposed hybrid tabu search algorithms (TS-TCL, TS-CL, TS-IL) and asses the quality of the solutions identified by these algorithms in small size problems. The quality of a solution identified by a heuristic can be best quantified by its (percentage) deviation from the actual optimal solution to that problem instance. However, as discussed previously, the problems investigated in this dissertation are NP-hard in the strong sense, which imply that exact optimization methods such as the branch-and-bound technique that guarantee identifying the actual optimal solution would require an excessive amount of time and memory, especially for medium and large size problem instances. This is the very reason that we propose tabu search algorithms for solving the scheduling problems effectively in a timely manner. While small size problem instances can be solved optimally with branch-and-bound technique reasonably quickly, it is impractical to solve large size instances optimally. Therefore, we quantify the performance of a tabu search algorithm with respect to the best lower bound identified on the optimal solution as:

$$Deviation = \frac{TS\ value - LB}{LB}\% \qquad (1)$$

The lower bound is the tightest bound that has been identified by the branch-and-bound technique with CPLEX, LBBD, or LP relaxation methods. If the optimal solution is found, the performance of tabu search is quantified by:

$$Deviation = \frac{TS\ value - Optimal\ Solution}{Optimal\ Solution}\% \qquad (2)$$

In case that the optimal solution is found, the performance of lower bounding methods can be quantified by

$$Deviation = \frac{Optimal\ Solution - LB}{Optimal\ Solution}\% \qquad (3)$$

The primary objectives of this chapter are as follows:

1. To quantify the performance of the tabu search algorithms with respect to the proposed lower bounds/optimal solution.
2. To quantify the performance of the proposed lower bounding methods with respect to the actual optimal solutions (for the problems for which the actual optimal solution can be identified).
3. To determine whether the three tabu search algorithms are statistically different in the quality and the time of the solutions they identify. If they are different, to determine which of the three tabu search algorithms is (are) the best.
4. To assess the effect of the labor flexibility, machine flexibility and scenario on the solution.

7.1. Data Generation

The data generation method used in this dissertation to test the computational performance of the developed search algorithms is based on previous study by Logendran and Subur [30]. The instances are categorized into three structures of small, medium and large. The number of jobs in small, medium and large size problems are generated from a uniform distribution in [3, 5], [6, 8], and [9,11], respectively. There are 3 stages and 2 skills in small, 4 stages and 3 skills in medium and 5 stages and 4 skills in large size problems. The sizes are identified in a way that branch-and-bound algorithms can solve the small size problems in less than 8 hours optimally and search algorithms can identify a reasonable solution for the large size problems in less than 8 hours. The total number of units of labor is based on labor flexibility. In problems with low labor flexibility there is one unit of labor for each skill level and the number of labor increases as labor flexibility increases. In small size problems there are 2 units of labor in low, 3 units of labor in medium and 4 units of labor in high labor flexibility. In medium size problems there are 3 units of labor in low, 4 units of labor in medium and 5 units of labor in high labor flexibility, and in large size problems there are 4 units of labor in low, 5 units of labor in medium and 6 units of labor in high labor flexibility. The assignment of labor to different skill levels is somewhat random. First of all one unit of labor is assigned to each skill level, and if there is an excess, a random number in [0, 1] is generated for the additional labor. In small problems the number of labor that has a value less than 1/2 and more than 1/2, are counted. The largest number of these counts is the number of additional labor required for skill level 1 and the smallest value is the number of additional labor required for skill level 2. In medium size problems, the number of labor that has a value less than 1/3, more than 2/3 and between 1/3 and 2/3, are counted. The largest number of these counts is the number of additional labor required for skill level 1, the smallest value is the number of additional labor required for skill level 3, and the one in between is the additional labor required for skill level 2. The same process is used for large problems as well; in large problems the number of labor that has a value less than 1/4, between than 1/4 and 2/4, between 2/4 and 3/4, and more than 3/4, are counted. The numbers are then sorted, and the largest number in the number of additional labor required for skill level 1 and the smallest number is the number of additional labor for skill level 4.

The machine arrangement of this problem is unrelated-parallel machines. The number of machines in each stage can vary between 1 to 3. There are three different machine flexibilities introduced in this research. If machine flexibility is low 2/3 of stages have single machine arrangement and 1/3 have multiple (2 or 3) machines. The decision between 2 or 3 machine is made randomly. In medium flexibility 1/3 of stages have single machines and 2/3 of stages have multiple machines and in high machine flexibility all stages have multiple machines. Three levels of capability are considered for the machines (most, medium and least capable). Three numbers are generated from a uniform distribution between [1, 5] as machine capability coefficients (α_i). The machine that has the smallest number is the most capable, the one with the largest number is the least capable and the one in between is the medium capable machine. If there are only two machines only two levels of capability (most and medium) exist. In case that the random number generated for two machines are equal it means that those machines are identical.

If all three random numbers are equal, all three machines are identical. In generating the machine arrangement at least one of the stages has to have an unrelated-parallel machine arrangement. If this did not happen, the machine arrangement is ignored and data generation process is repeated. Each machine can process different percentage of the jobs (β_i), depending on its capability. β_i is equal to 85, 70 and 50 for the most capable to least capable machine, respectively. For each job a random uniform number (RN) between 1 and 100 is generated. If RN> β_i then the job cannot be processed on that machine; otherwise its run time is generated from a uniform distribution in [2*α_i+2, 2*α_i+40]. The job setup time is generated uniformly between [1, 40]. There is a possibility that none of the machines are capable of processing the job, in this case job is allowed to skip that stage. Labor skill levels are assigned to job setups randomly from a uniform distribution of U[1, *p*] where *p* is the maximum number of skills, and assignment of skill levels to job runs is randomly chosen from a uniform distribution of U[1, *sp*], where *sp* is the skill level that has been assigned to that job's setup. The reason is that performing setups are harder than performing runs and they require higher skill level or the same skill level as runs.

As noted before there is a weight assigned to each job, and the weight is generated from U[1,3] where 3 shows the most important and 1 shows the least important job. Three different scenarios are also considered in this research for the entire collection of generated examples. The values of α and β are set to 0.7-0.3 to show that the producer is more important than customers, 0.5- 0.5 to show that both the producer and customers have equal importance, and 0.3-0.7 to show that customers are more important than the producer.

Both job release times and machine availability times are generated from a Poisson process with mean equal to λ. The value of λ depends on setup and run times and also the number of parallel machines in each stage. In this study $\lambda=1/20$. Most of the previous studies used Uniform distribution to generate these values and then accumulate the generated values to simulate the nature of job release times. Using accumulative Uniform distribution can be right if we are dealing with a single machine problem. Because of the memoryless property of Exponential distribution there is no need to accumulate the generated values like in Uniform distribution. The memoryless property gives the Exponential distribution the capability to simulate job release times/machine availability times in all of the structures with no more processing. Listed below is the Exponential distribution with mean of $1/\lambda$.

$$P(X > t) = e^{(-\lambda)t} \qquad (4)$$

To generate the desired value, first a random variable from a uniform distribution should be generated and then from that release times/machine availability times are generated using the following formula:

$$t = -\frac{1}{\lambda} Ln(\text{Random value}) \qquad (5)$$

In hybrid flowshop some justifications should be applied to evaluate the machine availability times for those machines in stage 2 and after. Note that there is no need to justify job release times. There are two different cases. In the first case there is only one machine at the current stage and in the second case there is more than one machine. If a_{it} is the machine *i* availability time at stage *t*, $\bar{s}_{iRt}$ is the average setup time of the job following the reference job on machine *i* at stage *t*, $\bar{p}_{it}$ is the average run time on machine *i* at stage *t* and a_{it}^{adj} is the adjusted machine *i* availability time at stage *t*, we have:

Case 1: Many-to-one: In this case there is one machine at current stage *t* and many machines at the previous stage *t-1*. The arrangement is shown is Figure 17. In this case the adjusted machine availability time is:

General Model (Many-to-one):

$$a_{1t}^{adj} = \max\{\textstyle\sum_{i'} a_{i't-1}^{adj}, a_{1t} + \bar{s}_{1Rt}\} + \bar{p}_{1t},\ \ i' = 1,2,\dots,m_{t-1} \qquad (6)$$

Special case (one-to-one):

$$a_{1t}^{adj} = \max\{a_{1t-1}^{adj}, a_{1t} + \bar{s}_{1Rt}\} + \bar{p}_{1t} \qquad (7)$$

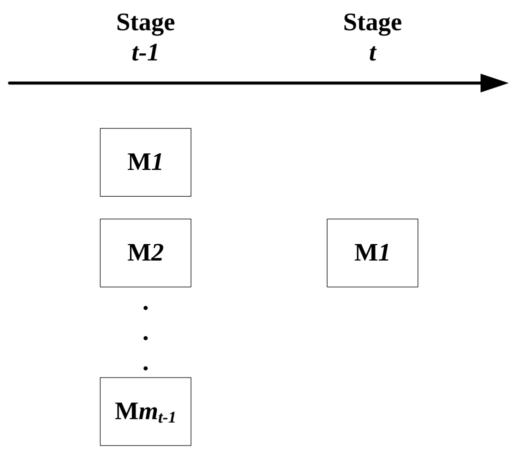

Figure 17. Many-to-one machine arrangement

Case 2: Many-to-many: In this case there are many machines at current stage t and many machines at previous stage t-1. This arrangement is shown in Figure 18. The adjusted machine availability time is evaluated as follows:

General model (many-to-many)

$$a_{it}^{adj} = \max\left\{\frac{m_{t-1}}{m_t}\max_{i'}\{a_{i't-1}^{adj}\}, a_{it} + \bar{s}_{iRt}\right\} + \bar{p}_{it},\ \ i = 1,2,\dots,m_t, i' = 1,2,\dots,m_{t-1}.$$

(7)

Special model (one-to-many)

$$a_{it}^{adj} = \max\left\{\frac{1}{m_t}a_{1t-1}^{adj}, a_{it} + \bar{s}_{iRt}\right\} + \bar{p}_{it},\ \ i = 1,2,\dots,m_t. \qquad (8)$$

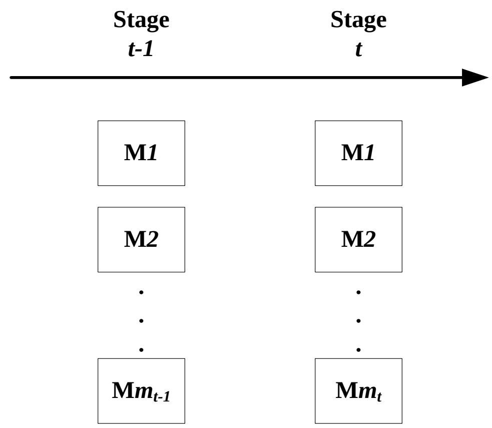

Figure 18. Many-to-many machine arrangement

Two factors, namely the due date range (R) and due date tightness (τ), are used to generate the due dates. There are three different ranges from tight to normal to wide (*R=0.8, 0.5 and 0.2*, respectively) used and the due date tightness is varied from tight to normal to wide (*τ=0.8, 0.5 and 0.2,* respectively). Due date tightness is defined as $\tau = 1 - \bar{d}/C_{max}$, where $\bar{d}$ is the average due date and C_{max} is the maximum expected completion time. The due date range is evaluated as $R = (d_{max} - d_{min})/C_{max}$, where d_{max} and d_{min} are the maximum and the minimum due date. The due date is then generated from a composite uniform distribution, with probability of τ the due date is from $U[\bar{d} - R\bar{d}, \bar{d}]$ and with probability (*1- τ)* it is from $U[\bar{d}, \bar{d} + (C_{max} - \bar{d})R]$. The maximum estimated completion time at each stage t is evaluated with

$$C_{\max t} = \sum_{j=1}^{n} max\left[r'_{jt}, a_{it} + (\delta \times savg_{ijt})\right] + p_{ijt}/m_{jt}, \quad (9)$$

where m_{jt} is the number of machines that can process job j at stage t and r'_{jt} is the updated release time of job j in stage t. Because this is a hybrid flowshop, the job release time is updated after the first stage. The updated release time of job j in stage t+1 is evaluated with

$$r'_{j\,1} = r_j$$
$$r'_{j\,t+1} = max\left[r'_{jt}, a_{it} + (\delta \times savg_{ijt})\right] + p_{ijt}, \quad (10)$$

In this calculation the average setup time is used. In reality the best schedule tends to use the smallest setup times in transferring from one job to another, so the average setup time adjuster (δ) is introduced. For evaluating δ, the coefficient of variation for each job on each machine is calculated with $CV = s/\bar{x}$, where s: standard deviation and $\bar{x}$: mean. If the problem was sequence-independent, then *CV*=0 and the adjuster would be equal to 1. Thus *CV*=0.01 is set equivalent to $\delta = 0.9$ and *CV*=1.0 is set equivalent to $\delta = 0.1$. A linear relationship to find the value of δ from *CV* is used as shown in Figure 19. If the *CV*s evaluated for a machine were bigger than 1, the biggest value should be set equal to 1 and the other *CV*s should be normalized. The maximum estimated completion time (C_{max}) is evaluated as $C_{max} = C_{\max T}$, where *T* is the index for the last stage.

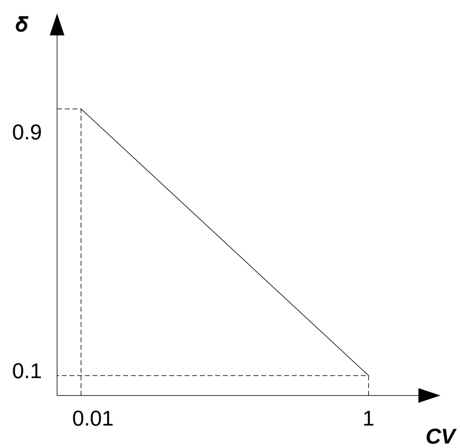

Figure 19. Relation between CV and δ

7.2. Experimental Analysis

In this chapter 81 examples are generated in small, medium and large sizes in three blocks for all labor and machine flexibility levels and in three different scenarios (3 labor flexibilities *3 machine flexibilities *3 blocks *3 scenarios=81). As explained before, there are three different labor and machine flexibilities (high, medium and low) and three different scenarios of 30-70%, which shows 30% importance of producer’s costs and 70% importance of customers’ costs, 50-50% and finally 70-30%. The blocks are differentiated from each other based on different number of jobs in them. For example, all examples in block 3 with small sizes have 5 jobs. The examples all have tight due dates and normal ranges. Figure 20 shows the arrangement of the examples in each size. In total 243 different examples were considered in this study. The characteristics of examples in small, medium and large sizes are reported in Tables 7.1-7.3.

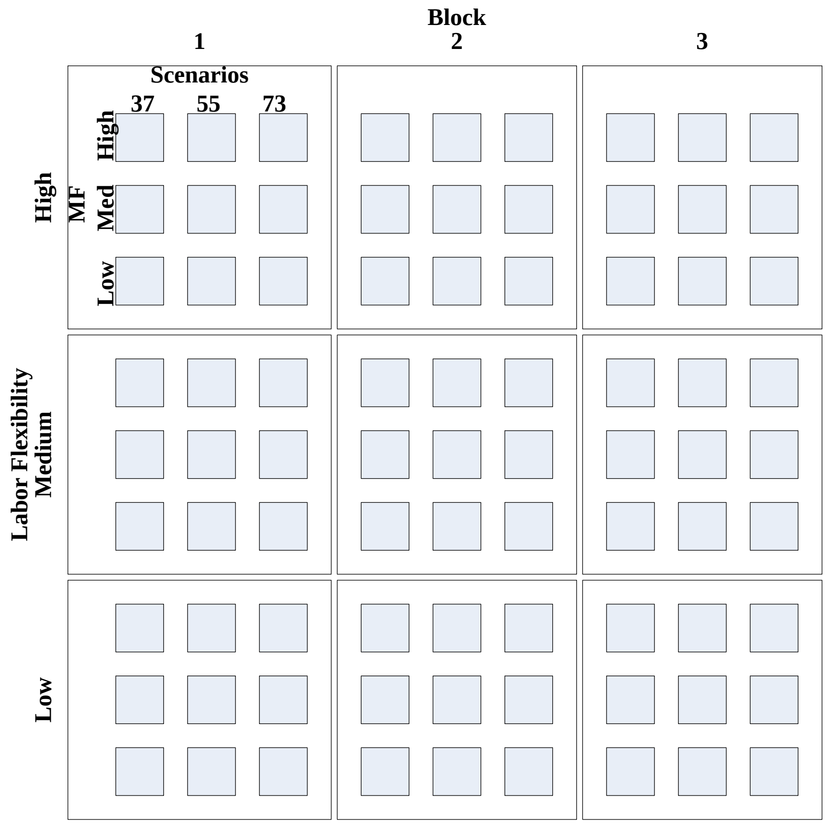

Figure 20. Arrangement of examples in small, medium or large size problems
* Scenarios 37, 55 and 73 are abbreviations for 30-70%, 50-50% and 70-30%, respectively.

Table 7.1 Examples characteristics in small size problems

Size	Block	Labor Flexibility	Machine Flexibility	# Jobs	# Stages	# Machines	# Skills	# Labors	# binary Variables	# Constraints
Small	1	High	High	3	3	8	2	4	462	988
Small	1	High	Medium	3	3	7	2	4	318	696
Small	1	High	Low	3	3	4	2	4	332	688
Small	1	Medium	High	3	3	8	2	3	315	687
Small	1	Medium	Medium	3	3	7	2	3	248	563
Small	1	Medium	Low	3	3	4	2	3	165	371
Small	1	Low	High	3	3	8	2	2	167	409
Small	1	Low	Medium	3	3	7	2	2	238	562
Small	1	Low	Low	3	3	4	2	2	124	297
Small	2	High	High	4	3	7	2	4	348	769
Small	2	High	Medium	4	3	6	2	4	689	1441
Small	2	High	Low	4	3	4	2	4	380	794
Small	2	Medium	High	4	3	7	2	3	552	1208
Small	2	Medium	Medium	4	3	6	2	3	170	393
Small	2	Medium	Low	4	3	4	2	3	318	697
Small	2	Low	High	4	3	7	2	2	268	637
Small	2	Low	Medium	4	3	6	2	2	282	685
Small	2	Low	Low	4	3	4	2	2	178	436
Small	3	High	High	5	3	7	2	4	756	1637
Small	3	High	Medium	5	3	6	2	4	932	1965
Small	3	High	Low	5	3	4	2	4	318	728
Small	3	Medium	High	5	3	7	2	3	668	1473
Small	3	Medium	Medium	5	3	6	2	3	810	1755
Small	3	Medium	Low	5	3	4	2	3	297	684
Small	3	Low	High	5	3	7	2	2	381	916
Small	3	Low	Medium	5	3	6	2	2	469	1133
Small	3	Low	Low	5	3	4	2	2	325	789

Table 7.2 Examples characteristics in medium size problems

Size	Block	Labor Flexibility	Machine Flexibility	# Jobs	# Stages	# Machines	# Skills	# Labors	# binary Variables	# Constraints
Medium	1	High	High	6	4	11	3	5	3420	7132
Medium	1	High	Medium	6	4	8	3	5	548	1268
Medium	1	High	Low	6	4	8	3	5	1130	2483
Medium	1	Medium	High	6	4	11	3	4	2008	4317
Medium	1	Medium	Medium	6	4	8	3	4	1077	2371
Medium	1	Medium	Low	6	4	8	3	4	1282	2766
Medium	1	Low	High	6	4	11	3	3	1103	2581
Medium	1	Low	Medium	6	4	8	3	3	760	1801
Medium	1	Low	Low	6	4	8	3	3	799	1876
Medium	2	High	High	7	4	11	3	5	2397	5124
Medium	2	High	Medium	7	4	8	3	5	1909	4118
Medium	2	High	Low	7	4	7	3	5	1649	3542
Medium	2	Medium	High	7	4	11	3	4	2183	4750
Medium	2	Medium	Medium	7	4	8	3	4	1311	2870
Medium	2	Medium	Low	7	4	7	3	4	1391	3082
Medium	2	Low	High	7	4	11	3	3	1618	3674
Medium	2	Low	Medium	7	4	8	3	3	1116	2562
Medium	2	Low	Low	7	4	7	3	3	651	1589
Medium	3	High	High	8	4	9	3	5	2351	5267
Medium	3	High	Medium	8	4	8	3	5	2613	5585
Medium	3	High	Low	8	4	8	3	5	1574	3509
Medium	3	Medium	High	8	4	9	3	4	2884	6183
Medium	3	Medium	Medium	8	4	8	3	4	2853	6070
Medium	3	Medium	Low	8	4	8	3	4	1742	3811
Medium	3	Low	High	8	4	9	3	3	1649	3797
Medium	3	Low	Medium	8	4	8	3	3	923	2198
Medium	3	Low	Low	8	4	8	3	3	904	2144

Table 7.3 Examples characteristics in large size problems

Size	Block	Labor Flexibility	Machine Flexibility	# Jobs	# Stages	# Machines	# Skills	# Labors	# binary Variables	# Constraints
Large	1	High	High	9	5	14	4	6	7852	16453
Large	1	High	Medium	9	5	11	4	6	5571	11860
Large	1	High	Low	9	5	8	4	6	2849	6128
Large	1	Medium	High	9	5	14	4	5	5982	12773
Large	1	Medium	Medium	9	5	11	4	5	3617	7906
Large	1	Medium	Low	9	5	8	4	5	2085	4631
Large	1	Low	High	9	5	14	4	4	3058	7032
Large	1	Low	Medium	9	5	11	4	4	2459	5685
Large	1	Low	Low	9	5	8	4	4	1538	3583
Large	2	High	High	10	5	12	4	6	6949	14685
Large	2	High	Medium	10	5	10	4	6	4559	9881
Large	2	High	Low	10	5	7	4	6	2111	4655
Large	2	Medium	High	10	5	12	4	5	5886	12870
Large	2	Medium	Medium	10	5	10	4	5	3476	7564
Large	2	Medium	Low	10	5	7	4	5	1865	4137
Large	2	Low	High	10	5	12	4	4	3057	7036
Large	2	Low	Medium	10	5	10	4	4	4559	9881
Large	2	Low	Low	10	5	7	4	4	1603	3871
Large	3	High	High	11	5	13	4	6	10018	21269
Large	3	High	Medium	11	5	12	4	6	7697	16486
Large	3	High	Low	11	5	8	4	6	3691	7830
Large	3	Medium	High	11	5	13	4	5	4963	10838
Large	3	Medium	Medium	11	5	12	4	5	8180	17481
Large	3	Medium	Low	11	5	8	4	5	5500	11779
Large	3	Low	High	11	5	13	4	4	3771	8652
Large	3	Low	Medium	11	5	12	4	4	3371	7718
Large	3	Low	Low	11	5	8	4	4	1612	3849

Each example is solved by three developed lower bounding methods (LBBD, F-LPR, and P-LPR) and with the traditional branch-and-bound (B&B) method. In addition, they all were run by search algorithms, TS-CL and TS-IL, for finding a good upper bound. Small size examples were run by TS-TCL as well. TS-TCL cannot be used for medium and large size problems due to its excessive computation time. This issue will be further discussed in this chapter. Therefore total number of runs is 1485. All these runs were performed on Intel Core 2 Duo P8800 (2.66GHz/1066MHz FSB/3M L2 Cache) with 4GB, DDR3, 1067 MHz 2 Dimm RAM. The lower bounding methods (LBBD, F-LPR and P-LPR) and B&B were coded with Visual C# 2008 [103] and used CPLEX 12.2 [91] for solving the problems. The search algorithms were coded in Visual C# 2008. All of the lower bounding methods, herein after also referred to simply as methods, and algorithms will be terminated after 8 hours (28800 s) if they have not been terminated sooner.

LBBD, F-LPR and P-LPR try to find a good lower bound while search algorithms such as TS-TCL, TS-CL and TS-IL try to find a good upper bound. Branch-and-bound method tries to solve the problem optimally and as the problems are NP-hard this method is not capable to find the optimal solution in all instances and therefore it reports its best lower and upper bound. CPLEX or any other mathematical solvers use B&B and are the most reliable software for solving mathematical programming problems. However, when the size of the problems grows in NP-hard problems they can take a long time in order to report an optimal solution or in some cases even a good solution. The problem that we are addressing in this dissertation is strongly NP-hard and the number of its binary variables and constraints increases exponentially as the size of problem grows. The complexity of the problem is so high that B&B cannot find a good solution for the problem or even if it finds a good solution, an exceedingly long time should be invested in order to get that solution. Our goal here is to find a more effective and efficient way to find the solution to this problem than B&B.

In order to identify if there is a statistically significant difference between the effectiveness and efficiency of the lower bounding methods, search algorithms and the B&B an experimental analysis is designed. Here a nested-factorial design with blocking is used [104]. The reason that a nested design is chosen rather than a factorial design is because the levels of machine flexibilities are similar but not identical for different levels of labor flexibilities. In this case the levels of machine flexibilities are nested under the levels of labor flexibilities. However, levels of methods (lower bounding or search algorithms) are arranged in a factorial layout. The blocking is based on the blocks that exist in the examples. The general model is:

$$y_{ijklm} = \mu + \tau_i + \beta_j + \gamma_{k(j)} + \delta_l + \rho_m + (\tau\beta)_{ij} + (\tau\gamma)_{ik(j)} + (\tau\delta)_{il} + (\beta\delta)_{jl} + (\gamma\delta)_{k(j)l} + (\tau\beta\delta)_{ijl} + (\tau\gamma\delta)_{ik(j)l} + \epsilon_{(ijkm)} \quad \begin{cases} i = 1,2,\dots,a \\ j = 1,2,3 \\ k = 1,2,3 \\ l = 1,2,3 \\ m = 1,2,3 \end{cases} \tag{11}$$

Where μ is the overall mean effect, τ_i represents the *i*th method (lower bounding methods or search algorithms), β_j is the effect of *j*th level of labor flexibility, $\gamma_{k(j)}$ is the effect of the *k*th machine flexibility within the *j*th level of labor flexibility, δ_l represents the effect of *l*th scenario, ρ_m represents the effect of *m*th block, $(\tau\beta)_{ij}$ is the effect of interaction between algorithm and labor flexibility, $(\tau\gamma)_{ik(j)}$ is the algorithm and machine flexibility within labor flexibility interaction, $(\tau\delta)_{il}$ is the effect of interaction between method and scenarios,$(\beta\delta)_{jl}$ is the effect of interaction between labor flexibility and scenario, $(\gamma\delta)_{k(j)l}$ is the machine flexibility within labor flexibility and scenario interaction, $(\tau\beta\delta)_{ijl}$ is the effect of interaction between method, labor flexibility and scenario, $(\tau\gamma\delta)_{ik(j)l}$ is the effect of interaction between method, machine flexibility within labor flexibility and scenario, and $\epsilon_{(ijkm)}$ is the error.

STATGRAPHICS [105] is used to perform statistical analysis. The results and statistical analysis for effectiveness and efficiency of lower bounding methods and search algorithms are provided in the following sub sections for each problem size.

7.2.1. Results and Analysis for Small Size Problems

In small size problems the optimal solution is obtained with branch-and-bound method and the performance assessment of search algorithms and lower bounding methods are based on optimal solution and so the percentage deviation that is provided is real because the optimal solution is known and the deviation is not reported based on a lower bound (which may be really far away from the optimal solution). The detail results for small examples are provided in Appendix B.

7.2.1.1. Effectiveness of lower bounding methods

As stated before B&B was capable to find the optimal solution in all small instances and therefore its deviation from the optimal solution is 0%, the lower bound found from LBBD, F-LPR and P-LPR have an average deviation of 1.34%, 1.34% and 1.41%, respectively as shown in Table 7.4.

Table 7.4 Lower bounding methods deviation in small examples

Labor Flexibility	Machine Flexibility	B&B	LBBD	F-LPR	P-LPR	Average
High	High	0.00%	0.00%	0.82%	0.82%	**0.41%**
	Medium	0.00%	0.00%	0.09%	0.09%	**0.04%**
	Low	0.00%	0.00%	0.00%	0.00%	**0.00%**
	Average	**0.00%**	**0.00%**	**0.30%**	**0.30%**	**0.15%**
Medium	High	0.00%	1.44%	5.75%	6.17%	**3.34%**
	Medium	0.00%	0.00%	4.05%	4.05%	**2.03%**
	Low	0.00%	0.00%	1.23%	1.23%	**0.62%**
	Average	**0.00%**	**0.48%**	**3.68%**	**3.82%**	**1.99%**
Low	High	0.00%	5.28%	0.13%	0.13%	**1.39%**
	Medium	0.00%	5.37%	0.00%	0.22%	**1.40%**
	Low	0.00%	0.00%	0.00%	0.00%	**0.00%**
	Average	**0.00%**	**3.55%**	**0.04%**	**0.12%**	**0.93%**
Average		**0.00%**	**1.34%**	**1.34%**	**1.41%**	**1.02%**

The results show a very good performance of methods in finding a lower bound. The results also show that LBBD quality decreases as the labor flexibility decreases. In other words, it performed better in high labor flexibility than medium, than low with deviations of 0%, 0.48%, and 3.55%, respectively. No such trend is seen in either F-LPR or P-LPR. They both performed better in low flexibility (0.04% and 0.12%) followed by high labor flexibility (0.30% and 0.30%) and have their worst performance in medium labor flexibility (3.68% and 3.82%). Also it seems that the effectiveness of lower bounding methods increases as the machine flexibility decreases.

The experimental analysis revealed that there is a significant difference between the quality of solutions found by different lower bounding methods including branch-and-bound. In addition there is a difference between the quality of solutions in different labor flexibility and quality differs when using different machine flexibility within labor flexibility. Also, the interaction between lower bounding methods and labor flexibility is statistically significant. The results show that scenario does not have any correlation with the quality of lower bound. The ANOVA is shown in Table 7.5.

Table 7.5 ANOVA for LB deviations in small examples

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Block	45100.4	2	22550.2	0.26	0.7704
Method	1.14E+06	3	378836	4.39	0.0051
Labor	1.85E+06	2	924162	10.7	0
Machine(Labor)	1.84E+06	6	306326	3.55	0.0023
Scenario	44105.3	2	22052.6	0.26	0.7749
Method*Labor	4.72E+06	6	787266	9.12	0
Method*Machine(Labor)	2.11E+06	18	117099	1.36	0.1562
Method*Scenario	29457	6	4909.49	0.06	0.9993
Labor*Scenario	60249.5	4	15062.4	0.17	0.9514
Scenario*Machine(Labor)	175255	12	14604.6	0.17	0.9993
Method*Labor*Scenario	155999	12	12999.9	0.15	0.9996
Method*Scenario*Machine(Labor)	154836	36	4301.01	0.05	1
Residual	1.85E+07	214	86355.5		
Total (corrected)	3.08E+07	323			

To determine which lower bounding methods perform better, Tukey's test was used. This method was preferred over the Fisher least significant difference (LSD) method as it is more conservative [104]. As the interaction between the factors is significant, the comparison between the means of solutions may be obscured by the interactions. So the Tukey's test for the means of one factor is performed by fixing the other factor at each of its level.

The test indicates that machine flexibility does not affect the quality of lower bound in high and low labor flexibility, however in medium labor flexibility, lower bounding methods performed better in low machine flexibility than high machine flexibility. Table 7.6 shows the summary result.

Table 7.6 Multiple comparison on machine flexibility levels within labor flexibility

Labor flexibility	Machine flexibility
High	-
Medium	Low>> High
Low	-

As observed in Table 7.4, statistical analysis approved that LBBD quality in high and medium labor flexibilities is better than low labor flexibilities. However, there is no significant difference between its performance in high and medium labor flexibilities. The quality of solutions with F-LPR and P-LPR are better in problems with high and low labor flexibilities than medium labor flexibility. Table 7.7 shows the summary result.

Table 7.7 Multiple comparison on labor flexibility levels by fixing lower bounding methods

Lower bounding method	Labor flexibility
B&B	-
LBBD	High=Medium>>Low
F-LPR	High=Low>>Medium
P-LPR	High=Low>>Medium

Detail analysis shows that in high labor flexibility there is no significant difference between the quality of lower bounding methods and they are statistically no different than the optimal solution. In medium labor flexibility LBBD solutions are statistically no different than B&B (optimal solutions), however, both B&B and LBBD outperformed F-LPR and P-LPR. Interestingly, the results are opposite in low labor flexibility. In fact in low labor flexibility, F-LPR and P-LPR were capable to find a lower bound almost as good as the optimal solution and LBBD failed in finding a good solution. Finally, we can conclude that the combination of the developed lower bounding methods were capable to find a lower bound almost equal to the optimal solution in all cases on average. Table 7.8 shows the summary result.

Table 7.8 Multiple comparison on lower bounding methods by fixing labor flexibility

Labor flexibility	Method
High	-
Medium	B&B=LBBB>>F-LPR=P-LPR
Low	B&B=F-LPR=P-LPR>>LBBD

7.2.1.2. Efficiency of lower bounding methods

Now that the statistical analysis has revealed that the lower bounding methods are in fact effective in small examples, it is important to investigate if they are efficient too. Table 7.9 is a summary table that shows the time spent by each method in seconds.

Table 7.9 Lower bounding methods time in small examples

Labor Flexibility	**Machine Flexibility**	**B&B**	**LBBD**	**F-LPR**	**P-LPR**	**Average**
High	**High**	122	37	8	8	**44**
	Medium	48	9	2	2	**15**
	Low	0	0	0	0	**0**
	Average	**57**	**15**	**3**	**3**	**20**
Medium	**High**	2896	15144	44	28	**4528**
	Medium	11	35	4	3	**13**
	Low	4	5	6	5	**5**
	Average	**970**	**5061**	**18**	**12**	**1515**
Low	**High**	113	15270	301	332	**4004**
	Medium	144	19233	305	324	**5001**
	Low	17	5337	44	55	**1363**
	Average	**91**	**13280**	**217**	**237**	**3456**
Average		**373**	**6119**	**79**	**84**	**1664**

As shown in the table, LP relaxation methods are the fastest and LBBD is very slow. In fact, LP relaxations are in average 4.5 times faster than B&B and 72 times faster than LBBD. B&B is 16 times faster than LBBD. Also by looking at the times, it seems that the methods need more time when labor flexibility decreases. The average time spent by all lower bounding methods is 20, 1515, and 3456 seconds for high, medium, and low labor flexibilities, respectively. Also the results suggest that the lower bounding methods need more time when there is higher machine flexibility.

Statistical analysis shows that there is a significant difference between the efficiency of different lower bounding methods. Also it states that labor flexibility and machine flexibility are affecting the efficiency. The interaction between different lower bounding methods and labor flexibility as well as different lower bounding methods and machine flexibility within labor flexibility are also significant. Again scenario is not related to the efficiency. Table 7.10 shows the ANOVA.

Table 7.10 ANOVA for LB time in small examples

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Block	4.60E+08	2	2.30E+08	10.77	0
Method	2.15E+09	3	7.16E+08	33.51	0
Labor	6.41E+08	2	3.21E+08	15.01	0
Machine(Labor)	7.45E+08	6	1.24E+08	5.81	0
Scenario	2.97E+06	2	1.49E+06	0.07	0.9329
Method*Labor	1.80E+09	6	2.99E+08	14	0
Method*Machine(Labor)	1.60E+09	18	8.90E+07	4.16	0
Method*Scenario	2.09E+07	6	3.49E+06	0.16	0.9861
Labor*Scenario	1.96E+06	4	489103	0.02	0.999
Scenario*Machine(Labor)	2.34E+07	12	1.95E+06	0.09	1
Method*Labor*Scenario	1.56E+07	12	1.30E+06	0.06	1
Method*Scenario*Machine(Labor)	1.14E+08	36	3.16E+06	0.15	1
Residual	4.57E+09	214	2.14E+07		
Total (corrected)	1.21E+10	323			

Tukey's test indicates that machine flexibility does not affect the efficiency of B&B, F-LPR and P-LPR, but it is a factor for LBBD. When the labor flexibility is high, the machine flexibility does not affect the efficiency, however in medium labor flexibility, LBBD is more efficient in low and medium machine flexibilities than high machine flexibility. In low labor flexibility, LBBD is more efficient only in low machine flexibility. Table 7.11 shows the summary result.

Table 7.11 Multiple comparison on machine flexibility levels within labor flexibility by fixing lower bounding methods

Method	Labor flexibility	Machine flexibility
B&B	High	-
	Medium	-
	Low	-
LBBD	High	-
	Medium	Low=Medium>>High
	Low	Low>>Medium=High
F-LPR	High	-
	Medium	-
	Low	-
P-LPR	High	-
	Medium	-
	Low	-

Table 7.12 Multiple comparison on lower bounding methods by fixing machine flexibility within labor flexibility

Labor flexibility	Machine flexibility	Method
High	High	-
	Medium	-
	Low	-
Medium	High	B&B=F-LPR=P-LPR>>LBBD
	Medium	-
	Low	-
Low	High	B&B=F-LPR=P-LPR>>LBBD
	Medium	B&B=F-LPR=P-LPR>>LBBD
	Low	-

There is no difference between the efficiency of B&B, F-LPR and P-LPR at all. LBBD is also as efficient as the other methods when the labor flexibility is high, but in medium labor flexibility and high machine flexibility it is less efficient than the other methods, however, in medium and low machine flexibilities there is no significant difference between the efficiency of the methods. In low labor flexibility, when the machine flexibility is high and medium, LBBD is less efficient than the other methods, but there is no significant difference between the efficiency of methods when the machine flexibility is low. Table 7.12 shows the summary result.

Table 7.13 shows that after performing pair-wise comparison the efficiency of LBBD is better in high than medium and they are both better than low labor flexibility. Labor flexibilities do not affect the efficiency of other lower bounding methods.

Table 7.13 Multiple comparison on labor flexibility by fixing lower bounding methods

Method	Labor flexibility
B&B	-
LBBD	High>>Medium>>Low
F-LPR	-
P-LPR	-

7.2.1.3. Effectiveness of search algorithms

As shown in Table 7.14, B&B was able to find the optimal solution in all instances and thus its deviation is 0%. The tabu search algorithms all together were able to find solutions with only an average deviation of 0.38% from the optimal solution. The TS-TCL, TS-CL, and TS-IL deviations were recorded as 1.03%, 1.50%, and 4.50%, respectively. The deviations seem to be very close to the optimal solution and it shows the high effectiveness of the search algorithm in small instances. The labor and machine flexibilities' trend can be also observed in the effectiveness of search algorithms. The higher the labor flexibility, the better is the quality of solution, and the lower the machine flexibility, the better is the quality of solution.

The experimental analysis showed that there is a significant difference between the quality of solutions found by different search algorithms including branch-and-bound. In addition there is a difference between the quality of solutions in different labor flexibility, and quality differs when using different machine flexibility within labor flexibility. The interaction between search algorithms and labor flexibility is statistically significant. Also the interaction between search algorithms and machine flexibility within labor flexibility is statistically significant. The results show that scenario does not have any correlation with the quality of search algorithm. The ANOVA is shown in Table 7.15.

Table 7.14 Search algorithms deviation in small examples

Labor Flexibility	Machine Flexibility	B&B	TS	TS-TCL	TS-CL	TS-IL	Average
High	**High**	0.00%	0.15%	0.15%	2.32%	4.89%	**1.50%**
	Medium	0.00%	0.56%	0.78%	0.67%	0.56%	**0.52%**
	Low	**0.00%**	**0.00%**	**0.00%**	**0.66%**	**0.66%**	**0.26%**
	Average	0.00%	0.24%	0.31%	1.22%	2.04%	**0.76%**
Medium	**High**	0.00%	0.07%	0.13%	2.75%	4.54%	**1.50%**
	Medium	0.00%	0.17%	0.66%	0.30%	0.45%	**0.32%**
	Low	0.00%	0.00%	0.55%	0.67%	3.63%	**0.97%**
	Average	**0.00%**	**0.08%**	**0.45%**	**1.24%**	**2.87%**	**0.93%**
Low	**High**	0.00%	1.25%	3.63%	3.23%	14.98%	**4.62%**
	Medium	0.00%	1.22%	2.69%	2.37%	10.44%	**3.34%**
	Low	0.00%	0.00%	0.69%	0.54%	0.32%	**0.31%**
	Average	**0.00%**	**0.82%**	**2.34%**	**2.05%**	**8.58%**	**2.76%**
Average		**0.00%**	**0.38%**	**1.03%**	**1.50%**	**4.50%**	**1.48%**

Table 7.15 ANOVA for search algorithms deviations in small examples

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Block	2.23E+06	2	1.11E+06	8.24	0.0003
Method	1.03E+07	4	2.57E+06	19.08	0
Labor	3.32E+06	2	1.66E+06	12.31	0
Machine(Labor)	5.11E+06	6	852491	6.32	0
Scenario	65652.8	2	32826.4	0.24	0.7843
Method*Labor	4.43E+06	8	554343	4.11	0.0001
Method*Machine(Labor)	8.34E+06	24	347319	2.57	0.0001
Method*Scenario	293326	8	36665.7	0.27	0.9747
Labor*Scenario	172372	4	43093.1	0.32	0.8649
Scenario*Machine(Labor)	469955	12	39162.9	0.29	0.9906
Method*Labor*Scenario	323831	16	20239.4	0.15	1
Method*Scenario*Machine(Labor)	4.54E+06	48	94559.2	0.7	0.9311
Residual	3.62E+07	268	134951		
Total (corrected)	7.58E+07	404			

The Tukey's test shows that the machine flexibility trend that was observed in Table 7.14 is not statistically significant and only TS-IL algorithm performs better in low machine flexibility when labor flexibility is low. Table 7.16 shows the summary result.

Table 7.16 Multiple comparison on machine flexibility levels within labor flexibility by fixing search algorithms

Search algorithms	Labor flexibility	Machine flexibility
B&B	High	-
	Medium	-
	Low	-
TS	High	-
	Medium	-
	Low	-
TCL	High	-
	Medium	-
	Low	-
CL	High	-
	Medium	-
	Low	-
IL	High	-
	Medium	-
	Low	Low >>High=Medium

* TS is the best result from all three algorithms: TS-TCL, TS-CL, and TS-IL
* TCL is used instead of TS-TCL
* CL is used instead of TS-CL
* IL is used instead of TS-IL

Also TS, TS-TCL and TS-CL perform as well as B&B (optimal solution) in all labor flexibilities. However, TS-IL performs better in high and medium labor flexibility than in low labor flexibility. Table 7.17 shows the summary result.

Table 7.17 Multiple comparison on labor flexibility levels by fixing search algorithms

Search algorithm	Labor flexibility
B&B	-
TS	-
TCL	-
CL	-
IL	High=Medium>>Low

Detail analysis shows that in high and medium labor flexibility there is no significant difference between the quality of search algorithms and they are statistically no different from the optimal solution. In low labor flexibility TS, TS-TCL, and TS-CL solutions are statistically no different from B&B (optimal solution), however, they all outperformed TS-IL when the machine flexibility is high and medium. In general we can conclude that two out of three developed search algorithms were in average capable to find a solution almost equal to the optimal solution in all cases. Table 7.18 shows the summary result.

Table 7.18 Multiple comparison on search algorithms by fixing machine flexibility within labor flexibility

Labor flexibility	Machine flexibility	Search Algorithms
High	High	-
	Medium	-
	Low	-
Medium	High	-
	Medium	-
	Low	-
Low	High	B&B=TS=TCL=CL>>IL
	Medium	B&B=TS=TCL=CL>>IL
	Low	-

7.2.1.4 Efficiency of search algorithms

Now that the statistical analysis showed that the search algorithms are in fact effective in small examples, the investigation is extended to determine if they are efficient too. Table 7.19 is a summery table that shows the time spends by each method in seconds.

Table 7.19 Time spent by search algorithms in small examples

Labor Flexibility	Machine Flexibility	B&B	TS	TS-TCL	TS-CL	TS-IL	Average
High	**High**	122	25	18	13	1	**36**
	Medium	48	79	76	13	8	**45**
	Low	**0**	**9**	**9**	**1**	**0**	**4**
	Average	57	38	34	9	3	**28**
Medium	**High**	2896	382	382	25	2	**738**
	Medium	11	84	84	5	2	**37**
	Low	4	136	136	12	2	**58**
	Average	**970**	**201**	**201**	**14**	**2**	**278**
Low	**High**	113	838	838	78	18	**377**
	Medium	144	515	515	60	3	**247**
	Low	17	330	330	32	9	**144**
	Average	**91**	**561**	**561**	**57**	**10**	**256**
Average		**373**	**266**	**265**	**26**	**5**	**187**

As shown in the table TS-IL is the fastest with using on average 5 s followed by TS-CL with 26 s and TS-TCL with 265 s. B&B is the least efficient with 373 s. The labor and machine flexibility trend can also be observed in the results. The higher the labor flexibility, the faster are the search algorithms, and the lower the machine flexibility, the faster are the search algorithms.

Statistical analysis shows that there is a significant difference between the efficiency of different search algorithms. Also it states that labor flexibility and machine flexibility are affecting the efficiency. The interaction between different search algorithms and labor flexibility as well as different search algorithms and machine flexibility within labor flexibility are also significant. Again scenario is not related to the efficiency. Table 7.20 shows the ANOVA.

Table 7.20 ANOVA for search algorithms time in small examples

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Block	8.46E+06	2	4.23E+06	10.97	0
Method	8.57E+06	4	2.14E+06	5.56	0.0003
Labor	5.16E+06	2	2.58E+06	6.69	0.0015
Machine(Labor)	1.56E+07	6	2.59E+06	6.73	0
Scenario	137927	2	68963.4	0.18	0.8363
Method*Labor	1.71E+07	8	2.14E+06	5.56	0
Method*Machine(Labor)	3.80E+07	24	1.58E+06	4.11	0
Method*Scenario	2.37E+06	8	295663	0.77	0.6323
Labor*Scenario	1.51E+06	4	376287	0.98	0.4211
Scenario*Machine(Labor)	2.64E+06	12	219717	0.57	0.8656
Method*Labor*Scenario	3.00E+06	16	187425	0.49	0.9528
Method*Scenario*Machine(Labor)	1.06E+07	48	220141	0.57	0.9896
Residual	1.03E+08	268	385553		
Total (corrected)	2.16E+08	404			

Tukey's test indicates that machine flexibility does not affect the efficiency of search algorithms. However, B&B is more efficient in low and medium machine flexibilities than high when the labor flexibility is medium. Table 7.21 shows the summary result.

Table 7.21 Multiple comparison on machine flexibility levels within labor flexibility by fixing search algorithms

Search algorithms	Labor flexibility	Machine flexibility
B&B	High	-
	Medium	Low=Medium>>High
	Low	-
TS	High	-
	Medium	-
	Low	-
TCL	High	-
	Medium	-
	Low	-
CL	High	-
	Medium	-
	Low	-
IL	High	-
	Medium	-
	Low	-

There is no difference between the efficiency of search algorithms at all. However B&B is less efficient than search algorithms in medium labor and high machine flexibilities. Table 7.22 shows the summary result.

Table 7.22 Multiple comparison on search algorithms by fixing machine flexibility within labor flexibility

Labor flexibility	Machine flexibility	Search algorithms
High	High	-
	Medium	-
	Low	-
Medium	High	TS=TCL=CL=IL>>B&B
	Medium	-
	Low	-
Low	High	-
	Medium	-
	Low	-

Table 7.23 shows that after performing pair-wise comparison the efficiency of all search algorithms is the same for different labor flexibilities, however, B&B is more efficient in high and low than medium labor flexibility.

Table 7.23 Multiple comparison on labor flexibility by fixing search algorithms

Search algorithms	Labor flexibility
B&B	High=Low>>Medium
TS	-
TCL	-
CL	-
IL	-

The interesting observation is that there is no significant difference between the quality of TS-TCL and TS-CL, and the optimal solution, but they are both more efficient than the B&B in some circumstances. Although, TS-TCL is very effective and the statistical analysis did not show any statistically significant difference between the time spent by TS-TCL (265 s) and TS-CL (26 s), the algorithm had poor efficiency when it comes to medium and large size problems as we were expecting while developing this method. Therefore, we did not include this algorithm for further analysis in medium and large size problems.

7.2.2 Results and Analysis for Medium Size Problems

In medium size problems 20.99% of the examples were solved optimally. For the other 79.01% of the problems, comparisons are made with the best lower bound. The deviation between a lower bound and an upper bound in some cases can be nonrealistic and thus could degrade the real quality of the upper bound. When the optimal solution is not found, the upper bound is compared with the best lower bound and a percentage deviation is reported. Whenever the real optimal solution is close to the lower bound the reported deviation represents the real performance of the upper bound, however when the optimal solution is very close to the upper bound the deviation that is reported is not realistic and can be misleading. We are facing this problem in medium and large size problem as the optimal solution is not known. The detail results for medium examples are in Appendix C.

7.2.2.1. Effectiveness of lower bounding methods

In medium size problem, finding a good lower bound becomes essential as the optimal solution is not known. Table 7.24 shows a summary of average deviation between the lower bound and the best lower bound found from different methods. As shown in the table, P-LPR has the best performance with average deviation of 0.95%, followed by F-LPR with 3.86%, followed by LBBD with 5.73%. The B&B method reported 9.7% deviation from the best lower bound and has the least performance. The deviations show in fact that all of the developed lower bounding methods were capable of tightening the lower bound found by B&B. It is important to note that results in Table 7.24 are average of three different scenarios and therefore there are some cases that none of the methods are pointing to 0% deviation. For example, in high labor and high machine flexibility P-LPR method is capable of finding the best lower bound in all three different scenarios that are being tested. However, in high labor and medium machine flexibility none of the methods has a 0% deviation, which means that different methods were able to find the best lower bound in different scenarios.

Table 7.24 Lower bounding methods deviation in medium examples

Labor Flexibility	Machine Flexibility	B&B	LBBD	F-LPR	P-LPR	Average
High	High	9.34%	0.26%	4.36%	0.00%	**3.49%**
	Medium	7.18%	0.03%	5.65%	0.19%	**3.26%**
	Low	0.03%	0.88%	0.18%	0.45%	**0.38%**
	Average	**5.52%**	**0.39%**	**3.40%**	**0.21%**	**2.38%**
Medium	High	13.66%	1.58%	2.99%	1.28%	**4.88%**
	Medium	9.22%	2.13%	0.55%	1.58%	**3.37%**
	Low	3.84%	3.57%	10.90%	0.50%	**4.70%**
	Average	**8.91%**	**2.43%**	**4.81%**	**1.12%**	**4.32%**
Low	High	22.76%	16.37%	4.20%	1.65%	**11.25%**
	Medium	13.92%	11.44%	3.91%	1.53%	**7.70%**
	Low	7.35%	15.30%	1.95%	1.37%	**6.49%**
	Average	**14.68%**	**14.37%**	**3.35%**	**1.52%**	**8.48%**
Average		**9.70%**	**5.73%**	**3.86%**	**0.95%**	**5.06%**

The results also show that both LBBD and B&B have a hard time finding good quality solutions as the labor flexibility decreases. However, the quality of LP relaxation methods seems to be indifferent to labor flexibly.

The experimental analysis revealed that there is a significant difference between the quality of solutions found by different lower bounding methods. In addition there is a difference between the quality of solutions in different labor flexibility and quality differs when using different machine flexibility within labor flexibility. The interaction between lower bounding methods and labor flexibility and also the interaction between lower bounding methods and machine flexibility within labor flexibility are statistically significant. The results show that scenario does not have any correlation with the quality of lower bound. The ANOVA is shown in Table 7.25.

Table 7.25 ANOVA for LB deviations in medium examples

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Block	8.63E+06	2	4.31E+06	10.18	0.0001
Method	3.26E+07	3	1.09E+07	25.68	0
Labor	2.10E+07	2	1.05E+07	24.78	0
Machine(Labor)	7.04E+06	6	1.17E+06	2.77	0.013
Scenario	753049	2	376525	0.89	0.4127
Method*Labor	2.20E+07	6	3.67E+06	8.66	0
Method*Machine(Labor)	2.09E+07	18	1.16E+06	2.74	0.0003
Method*Scenario	1.23E+06	6	205807	0.49	0.8186
Labor*Scenario	1.04E+06	4	259466	0.61	0.6542
Scenario*Machine(Labor)	2.22E+06	12	184666	0.44	0.9477
Method*Labor*Scenario	1.15E+06	12	95507.9	0.23	0.9971
Method*Scenario*Machine(Labor)	3.07E+06	36	85316.6	0.2	1
Residual	9.07E+07	214	423721		
Total (corrected)	2.12E+08	323			

The results show that machine flexibility does not affect the quality of lower bounds. However, B&B performs better in low machine flexibility than high when the labor flexibility is low. Table 7.26 shows the summary result.

Table 7.26 Multiple comparison on machine flexibility levels within labor flexibility by fixing lower bounding methods

Method	Labor flexibility	Machine flexibility
B&B	High	-
	Medium	-
	Low	Low>>High
LBBD	High	-
	Medium	-
	Low	-
F-LPR	High	-
	Medium	-
	Low	-
P-LPR	High	-
	Medium	-
	Low	-

As predicted before the quality of LP relaxation is independent of labor flexibility. But both B&B and LBBD have better quality in high and medium than low labor flexibility. Table 7.27 shows the summary result.

Table 7.27 Multiple comparison on labor flexibility levels by fixing lower bounding methods

Method	Labor flexibility
B&B	High=Medium>>Low
LBBD	High=Medium>>Low
F-LPR	-
P-LPR	-

Table 7.28 Multiple comparison on lower bounding methods by fixing labor flexibility

Labor flexibility	Machine flexibility	Method
High	High	-
	Medium	-
	Low	-
Medium	High	P-LPR>>B&B
	Medium	-
	Low	-
Low	High	P-LPR>>B&B,LBBD F-LPR>>B&B
	Medium	P-LPR>>B&B
	Low	P-LPR,F-LPR>>LBBD

Further analysis shows that P-LPR performs better than B&B in medium labor and high machine flexibilities. In low labor flexibility, P-LPR performs better than both B&B and LBBD and F-LPR performs better than B&B in high machine flexibilities. Also P-LPR has a better quality than B&B in medium machine flexibility. Both P-LPR and F-LPR outperformed LBBD in low machine flexibility. Table 7.28 shows the summary result.

7.2.2.2. Efficiency of lower bounding methods

Statistical analysis showed that P-LPR is the most effective lower bounding method in the medium structure. Also the other important issue is the efficiency of the methods. Table 7.29 is a summery table that shows the time spent by each method in seconds.

As shown in the table, P-LPR is the most efficient method with average time of 14136 s, followed by F-LPR with 18515 s, B&B with 23350 s and LBBD with 25089 s. The interesting observation is that LP relaxation methods are the most effective and also the most efficient lower bounding methods.

Table 7.29 Lower bounding methods time in medium examples

Labor Flexibility	Machine Flexibility	B&B	LBBD	F-LPR	P-LPR	Average
High	High	28800	28800	23705	13333	**23660**
	Medium	19201	19200	12841	293	**12884**
	Low	9741	14590	988	157	**6369**
	Average	**19248**	**20863**	**12511**	**4594**	**14304**
Medium	High	28800	28800	28800	16687	**25772**
	Medium	19213	19208	10333	6659	**13853**
	Low	28800	28800	17536	16513	**22912**
	Average	**25604**	**25603**	**18890**	**13286**	**20846**
Low	High	28800	28800	25603	28800	**28001**
	Medium	20794	28800	22763	23183	**23885**
	Low	26002	28800	24066	21601	**25117**
	Average	**25199**	**28800**	**24144**	**24528**	**25668**
Average		**23350**	**25089**	**18515**	**14136**	**20273**

Statistical analysis shows that there is a significant difference between the efficiency of different lower bounding methods. Also it states that labor flexibility and machine flexibility are affecting the efficiency. The interaction between different lower bounding methods and labor flexibility is also significant. Again scenario is not related to the efficiency. Table 7.30 shows the ANOVA.

Table 7.30 ANOVA for LB time in medium examples

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Block	5.24E+09	2	2.62E+09	27.78	0
Method	5.95E+09	3	1.98E+09	21.01	0
Labor	7.03E+09	2	3.51E+09	37.24	0
Machine(Labor)	8.60E+09	6	1.43E+09	15.19	0
Scenario	7.56E+07	2	3.78E+07	0.4	0.6705
Method*Labor	1.74E+09	6	2.91E+08	3.08	0.0065
Method*Machine(Labor)	1.18E+09	18	6.57E+07	0.7	0.8124
Method*Scenario	6.71E+07	6	1.12E+07	0.12	0.9941
Labor*Scenario	2.56E+08	4	6.40E+07	0.68	0.6075
Scenario*Machine(Labor)	3.17E+08	12	2.64E+07	0.28	0.9919
Method*Labor*Scenario	4.04E+08	12	3.37E+07	0.36	0.9765
Method*Scenario*Machine(Labor)	6.80E+08	36	1.89E+07	0.2	1
Residual	2.02E+10	214	9.43E+07		
Total (corrected)	5.17E+10	323			

Tukey's test indicates that methods are more efficient in low and medium than high machine flexibility when labor flexibility is high. Also they are more efficient in medium machine flexibility and medium labor flexibility. Table 7.31 shows the summary result.

Table 7.31 Multiple comparison on machine flexibility levels within labor flexibility

Labor flexibility	Machine flexibility
High	Low=Medium>>High
Medium	Medium>>Low=High
Low	-

Analysis revealed that statistically, P-LPR is more efficient than B&B and LBBD in high and medium labor flexibility. Table 7.32 shows the summary result.

Table 7.32 Multiple comparison on lower bounding methods by fixing labor flexibility

Labor flexibility	Method
High	P-LPR>>B&B,LBBD
Medium	P-LPR>>B&B,LBBD
Low	-

Table 7.33 shows that P-LPR is more efficient in high than medium and in medium than low labor flexibility, and F-LPR is more efficient in high than low labor flexibility.

Table 7.33 Multiple comparison on labor flexibility by fixing lower bounding methods flexibility

Method	Labor flexibility
B&B	-
LBBD	-
F-LPR	High>>Low
P-LPR	High>>Medium>>Low

7.2.2.3. Effectiveness of search algorithms

As stated before, in the absence of an optimal solution, the deviations are obtained from the best lower bounding method and unfortunately this deviation can sometimes mask the good quality of the upper bound. Table 7.34 shows the percentage deviation of the upper bound found from search algorithms and B&B method with the best lower bound.

Table 7.34 Search algorithms deviation with best lower bound in medium examples

Labor Flexibility	Machine Flexibility	B&B	TS	TS-CL	TS-IL	Average
High	**High**	9.65%	12.10%	13.24%	15.18%	**12.54%**
	Medium	3.84%	5.45%	5.45%	8.89%	**5.91%**
	Low	3.45%	6.68%	6.69%	7.57%	**6.10%**
	Average	**5.65%**	**8.08%**	**8.46%**	**10.55%**	**8.18%**
Medium	**High**	21.40%	22.97%	23.25%	34.89%	**25.63%**
	Medium	5.72%	6.43%	6.47%	15.82%	**8.61%**
	Low	5.21%	7.52%	7.52%	15.26%	**8.88%**
	Average	**10.77%**	**12.31%**	**12.41%**	**21.99%**	**14.37%**
Low	**High**	30.83%	35.69%	36.63%	53.94%	**39.27%**
	Medium	10.35%	9.82%	11.04%	17.22%	**12.11%**
	Low	3.10%	7.84%	8.27%	14.39%	**8.40%**
	Average	**14.76%**	**17.78%**	**18.64%**	**28.52%**	**19.93%**
Average		**10.39%**	**12.72%**	**13.17%**	**20.35%**	**14.16%**

As shown in the table the quality of B&B, TS and TS-CL are almost the same and TS-IL is less effective than the other methods. Usually the upper bound found with the B&B method and using the mathematical solver such as CPLEX are very effective and although the problem is NP-hard and there is no guarantee that they can find the optimal solution within their time limit (8 hours), they are capable to provide a good upper bound. But still that upper bound is obtained after several hours of hours, which is usually not that appealing to practitioners in industry as they do not have the luxury of spending that much time. So usually the reason that industries do not go after mathematical solvers and B&B method is due to their inefficiency, rather than their effectiveness.

The complexity of proposed problems are so high that the goal here is not to show that the average deviation of search algorithms and the best lower bound is almost negligible, but to show that we can get the same effectiveness as the B&B method (the best solution that can be found by available solvers) without spending excessive amount of time. Although it is very encouraging to see small deviations between the search algorithm and the best lower bound, this seems to be almost impossible with the complexity that is observed in this problem. That is why Table 7.35 shows the average percentage deviation between the search algorithms and the best upper bound found.

As shown in the table, B&B deviation is 0.45%. The tabu search algorithms all together were able to find solutions with only an average deviation of 2.66%. The TS-CL and TS-IL deviations were recorded as 3.05% and 9.35%, respectively.

Table 7.35 Search algorithms deviation with best upper bound in medium examples

Labor Flexibility	Machine Flexibility	B&B	TS	TS-CL	TS-IL	Average
High	High	0.53%	2.71%	3.71%	5.55%	**3.12%**
	Medium	0.05%	1.60%	1.60%	4.87%	**2.03%**
	Low	0.00%	3.14%	3.15%	4.01%	**2.58%**
	Average	**0.19%**	**2.48%**	**2.82%**	**4.81%**	**2.58%**
Medium	High	0.17%	1.50%	1.73%	11.47%	**3.72%**
	Medium	0.68%	1.44%	1.47%	10.06%	**3.41%**
	Low	0.79%	3.01%	3.01%	10.49%	**4.33%**
	Average	**0.55%**	**1.98%**	**2.07%**	**10.68%**	**3.82%**
Low	High	0.00%	4.05%	4.73%	17.84%	**6.66%**
	Medium	2.23%	1.93%	3.06%	8.90%	**4.03%**
	Low	0.00%	4.57%	4.98%	10.94%	**5.12%**
	Average	**0.74%**	**3.52%**	**4.26%**	**12.56%**	**5.27%**
Average		**0.49%**	**2.66%**	**3.05%**	**9.35%**	**3.89%**

The experimental analysis showed that there is a significant difference between the quality of solutions found by different search algorithms. In addition there is a difference between the qualities of solutions in different labor flexibility. The interaction between search algorithms and labor flexibility is also statistically significant. The results show that scenario and machine flexibility do not have any correlation with the quality of search algorithm. The ANOVA is shown in Table 7.36.

Table 7.36 ANOVA for search algorithms deviations in medium examples

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Block	1.28E+06	2	640005	2.43	0.0902
Method	3.53E+07	3	1.18E+07	44.69	0
Labor	3.92E+06	2	1.96E+06	7.46	0.0007
Machine(Labor)	1.62E+06	6	270419	1.03	0.408
Scenario	1.10E+06	2	548126	2.08	0.127
Method*Labor	5.94E+06	6	989505	3.76	0.0014
Method*Machine(Labor)	4.00E+06	18	222062	0.84	0.6467
Method*Scenario	1.22E+06	6	203390	0.77	0.5917
Labor*Scenario	1.98E+06	4	496197	1.89	0.1139
Scenario*Machine(Labor)	2.88E+06	12	240071	0.91	0.5351
Method*Labor*Scenario	3.08E+06	12	256804	0.98	0.4725
Method*Scenario*Machine(Labor)	6.03E+06	36	167364	0.64	0.947
Residual	5.63E+07	214	263043		
Total (corrected)	1.25E+08	323			

Pair-wise comparisons show that the effectiveness of B&B, TS and TS-CL does not change with labor flexibility. However, TS-IL has a better quality in high than medium and low labor flexibilities. Table 7.37 shows the summary result.

Table 7.37 Multiple comparison on labor flexibility levels by fixing search algorithms

Search algorithm	Labor flexibility
B&B	-
TS	-
CL	-
IL	high>>Medium=Low

Detail analysis shows that the quality of TS and TS-CL is the same as B&B. However, they all have a better quality than TS-IL in medium and low labor flexibilities. Table 7.38 shows the summary result.

Table 7.38 Multiple comparison on search algorithms by fixing labor flexibility

Labor flexibility	Method
High	-
Medium	B&B=TS=CL>>IL
Low	B&B=TS=CL>>IL

7.2.2.4. Efficiency of search algorithms

Now that the statistical analysis showed that the effectiveness of the search algorithm TS-CL is as good as B&B, the efficiency becomes the next most important issue in attesting to the capability of the search algorithm. Table 7.39 is a summery table that shows the time spent by each search algorithm in seconds.

The result shows a huge difference between the efficiency of the search algorithms and the B&B. TS-IL is the most efficient algorithm that used on average 529 s, next TS-CL which spent 2295 s, the combined TS used 2314 s and B&B spent 23661 s. In fact TS-IL is 4 times more efficient than TS-CL and 44 times more efficient than B&B. However, one can argue that the quality of solution found by TS-IL is not as good as B&B and TS-CL. TS-CL is also 10 times more efficient than B&B and it has the same quality as B&B.

Table 7.39 Time spent by search algorithms in medium examples

Labor Flexibility	Machine Flexibility	B&B	TS	TS-CL	TS-IL	Average
High	**High**	28800	2225	2055	892	**8493**
	Medium	19201	2241	2241	82	**5942**
	Low	9741	1263	1263	348	**3154**
	Average	**19248**	**1910**	**1853**	**441**	**5863**
Medium	**High**	28800	3311	3311	171	**8898**
	Medium	19213	1930	1930	186	**5815**
	Low	28800	1483	1483	418	**8046**
	Average	**25604**	**2241**	**2241**	**259**	**7586**
Low	**High**	28800	2507	2507	294	**8527**
	Medium	20794	2985	2985	1464	**7057**
	Low	28800	2879	2878	903	**8865**
	Average	**26131**	**2790**	**2790**	**887**	**8150**
Average		**23661**	**2314**	**2295**	**529**	**7200**

Statistical analysis shows that there is a significant difference between the efficiency of different search algorithms. Also it states that labor flexibility and machine flexibility are affecting the efficiency. The interaction between different search algorithms and labor flexibility as well as different search algorithms and machine flexibility within labor flexibility are also significant. Again scenario is not related to the efficiency. Table 7.40 shows the ANOVA.

Table 7.40 ANOVA for search algorithms time in medium examples

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Block	1.23E+09	2	6.17E+08	22.33	0
Method	2.86E+10	3	9.54E+09	345.21	0
Labor	2.63E+08	2	1.31E+08	4.75	0.0095
Machine(Labor)	7.38E+08	6	1.23E+08	4.45	0.0003
Scenario	1.14E+07	2	5.70E+06	0.21	0.8138
Method*Labor	4.50E+08	6	7.49E+07	2.71	0.0147
Method*Machine(Labor)	1.80E+09	18	1.00E+08	3.62	0
Method*Scenario	6.61E+07	6	1.10E+07	0.4	0.8795
Labor*Scenario	1.46E+07	4	3.65E+06	0.13	0.9706
Scenario*Machine(Labor)	3.39E+07	12	2.82E+06	0.1	1
Method*Labor*Scenario	3.58E+07	12	2.98E+06	0.11	0.9999
Method*Scenario*Machine(Labor)	7.80E+07	36	2.17E+06	0.08	1
Residual	5.91E+09	214	2.76E+07		
Total (corrected)	3.92E+10	323			

Tukey’s test indicates that machine flexibility does not affect the efficiency of search algorithms. However, B&B is more efficient in low and medium machine flexibilities than high when the labor flexibility is high and is more efficient in medium machine flexibility than low and high when the labor flexibility is medium. Table 7.41 shows the summary result.

Further analysis shows that there is no statistical difference between TS, TS-CL and TS-IL efficiency and they are all more efficient than B&B in different levels of machine flexibility within labor flexibility, except that in low machine flexibility and high labor flexibility there is no significant difference between search algorithms and B&B. Table 7.42 summarizes the results.

Table 7.41 Multiple comparison on machine flexibility levels within labor flexibility by fixing search algorithms

Search algorithm	Labor flexibility	Machine flexibility
B&B	High	Low=Medium>>High
	Medium	Medium>>Low=High
	Low	-
TS	High	-
	Medium	-
	Low	-
TCL	High	-
	Medium	-
	Low	-
CL	High	-
	Medium	-
	Low	-
IL	High	-
	Medium	-
	Low	-

Table 7.42 Multiple comparison on search algorithms by fixing machine flexibility within labor flexibility

Labor flexibility	Machine flexibility	Search algorithm
High	High	IL=CL=TS>>B&B
	Medium	IL=CL=TS>>B&B
	Low	-
Medium	High	IL=CL=TS>>B&B
	Medium	IL=CL=TS>>B&B
	Low	IL=CL=TS>>B&B
Low	High	IL=CL=TS>>B&B
	Medium	IL=CL=TS>>B&B
	Low	IL=CL=TS>>B&B

Table 7.43 shows that after performing pair wise comparison the efficiency of all search algorithms is the same in different labor flexibilities, however, B&B is more efficient in high than in medium and low labor flexibilities.

Table 7.43 Multiple comparison on labor flexibility by fixing search algorithms

Method	Labor flexibility
B&B	High>>Medium=Low
TS	-
TCL	-
CL	-
IL	-

The results from medium structure show that TS-CL is as effective as B&B method was and is 10 times more efficient than B&B, which proves the power of the developed search algorithms.

7.2.3. Results and Analysis for Large Size Problems

None of the examples in large size problems were solved optimally and so all of the solutions are compared with the best lower bound and the issues discussed in medium size problems hold true for large problems as well. The detail results for large examples are in Appendix D.

7.2.3.1. Effectiveness of lower bounding methods

Table 7.44 shows a summary of average deviation between the lower bound and the best lower bound found from different methods. As shown in the table, LBBD has the best performance with average deviation of 2.51%, followed by F-LPR with 9.44%, followed by P-LPR with 1249%. The B&B method reported 26.32% deviation from the best lower bound and has the least performance. The deviations show in fact that all of the developed lower bounding methods were capable of tightening the lower bound found by B&B. Labor and machine flexibility trends cannot be seen easily in this table.

Table 7.44 Lower bounding methods deviation in large examples

Labor Flexibility	Machine Flexibility	B&B	LBBD	F-LPR	P-LPR	Average
High	High	22.99%	0.00%	16.95%	20.65%	**15.15%**
	Medium	24.23%	0.00%	14.78%	18.19%	**14.30%**
	Low	25.30%	0.60%	2.73%	5.21%	**8.46%**
	Average	**24.17%**	**0.20%**	**11.49%**	**14.69%**	**12.64%**
Medium	High	25.21%	2.97%	3.56%	16.46%	**12.05%**
	Medium	23.73%	0.72%	12.65%	15.94%	**13.26%**
	Low	26.67%	0.70%	12.40%	12.85%	**13.15%**
	Average	**25.21%**	**1.46%**	**9.54%**	**15.08%**	**12.82%**
Low	High	22.83%	7.76%	5.53%	5.66%	**10.45%**
	Medium	37.41%	7.30%	5.28%	2.69%	**13.17%**
	Low	28.50%	2.54%	11.05%	14.77%	**14.22%**
	Average	**29.58%**	**5.86%**	**7.29%**	**7.71%**	**12.61%**
Average		**26.32%**	**2.51%**	**9.44%**	**12.49%**	**12.69%**

The experimental analysis revealed that there is a significant difference between the quality of solutions found by different lower bounding methods. In addition there is a difference between the quality of solutions in different machine flexibilities within labor flexibility. The effect of labor flexibility factor is not significant on LB quality. The interaction between lower bounding methods and labor flexibility and also the interaction between lower bounding methods and machine flexibility within labor flexibility are statistically significant. The results show that scenario does not have any correlation with the quality of lower bound. The ANOVA is shown in Table 7.45.

Table 7.45 ANOVA for LB deviations in large examples

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Block	9.88E+06	2	4.94E+06	5.76	0.0037
Method	2.43E+08	3	8.10E+07	94.5	0
Labor	29355.6	2	14677.8	0.02	0.983
Machine(Labor)	1.26E+07	6	2.10E+06	2.45	0.026
Scenario	27700.2	2	13850.1	0.02	0.984
Method*Labor	2.09E+07	6	3.48E+06	4.06	0.0007
Method*Machine(Labor)	3.71E+07	18	2.06E+06	2.4	0.0016
Method*Scenario	3.47E+06	6	578063	0.67	0.6705
Labor*Scenario	2.01E+06	4	502981	0.59	0.6726
Scenario*Machine(Labor)	3.58E+06	12	298114	0.35	0.9789
Method*Labor*Scenario	2.32E+06	12	193431	0.23	0.997
Method*Scenario*Machine(Labor)	5.94E+06	36	165036	0.19	1
Residual	1.83E+08	214	857191		
Total (corrected)	5.24E+08	323			

The results show that machine flexibility does not affect the quality of lower bounds. However, P-LPR performs better in low machine flexibility than high and medium when the labor flexibility is high. Table 7.46 shows the summary result.

Further analysis shows that LBBD performs better than B&B and P-LPR in both high and medium machine flexibilities and LBBD, F-LPR and P-LPR have the same quality and they all have a better quality than B&B in low machine flexibility when labor flexibility is high. LBBD and F-LPR outperform B&B in high machine flexibility and LBBD's quality is better than B&B in medium and low machine flexibilities when labor flexibility is medium. Also LBBD, F-LPR and P-LPR perform the same and they all have a better quality than B&B in medium machine flexibility and LBBD has a better quality than B&B in low machine flexibility when labor flexibility is low. Table 7.47 shows the summary result.

Table 7.46 Multiple comparison on machine flexibility levels within labor flexibility by fixing lower bounding methods

Method	Labor flexibility	Machine flexibility
B&B	High	-
	Medium	-
	Low	-
LBBD	High	-
	Medium	-
	Low	-
F-LPR	High	-
	Medium	-
	Low	-
P-LPR	High	Low>>High=Medium
	Medium	-
	Low	-

Table 7.47 Multiple comparison on lower bounding methods by fixing labor flexibility

Labor flexibility	Machine flexibility	Method
High	High	LBBD>>B&B, P-LPR
	Medium	LBBD>>B&B, P-LPR
	Low	LBBD=F-LPR=P-LPR>>B&B
Medium	High	LBBD, F-LPR>>B&B
	Medium	LBBD>>B&B
	Low	LBBD>>B&B
Low	High	-
	Medium	P-LPR=F-LPR=LBBD>>B&B
	Low	LBBD>>B&B

7.2.3.2 Efficiency of lower bounding methods

Statistical analysis showed that LBBD is the most effective lower bounding method in the large structure. Also the other important issue is the efficiency of the methods. Table 7.48 is a summery table that shows the time spent by each method in seconds. As shown in the table, P-LPR is the most efficient method with average time of 26647 s, followed by F-LPR with 27881 s, B&B with 28800 s and LBBD with 28800 s.

Table 7.48 Lower bounding methods time in large examples

Labor Flexibility	Machine Flexibility	B&B	LBBD	F-LPR	P-LPR	Average
High	High	28800	28800	28800	28800	**28800**
	Medium	28800	28800	28800	28800	**28800**
	Low	28800	28800	26931	26485	**27754**
	Average	**28800**	**28800**	**28177**	**28028**	**28451**
Medium	High	28800	28800	28800	28800	**28800**
	Medium	28800	28800	28800	28800	**28800**
	Low	28800	28800	22400	11737	**22934**
	Average	**28800**	**28800**	**26667**	**23112**	**26845**
Low	High	28800	28800	28800	28800	**28800**
	Medium	28800	28800	28800	28800	**28800**
	Low	28800	28800	28800	28800	**28800**
	Average	**28800**	**28800**	**28800**	**28800**	**28800**
Average		**28800**	**28800**	**27881**	**26647**	**28032**

Statistical analysis shows that there is a significant difference between the efficiency of different lower bounding methods. Also it states that labor flexibility and machine flexibility are affecting the efficiency. The interaction between different lower bounding methods and labor flexibility and different lower bounding methods and machine flexibility within labor flexibility is also significant. Again scenario is not related to the efficiency. Table 7.49 shows the ANOVA.

Table 7.49 ANOVA for LB time in large examples

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Block	1.56E+08	2	7.81E+07	5.96	0.003
Method	2.53E+08	3	8.43E+07	6.43	0.0003
Labor	2.35E+08	2	1.17E+08	8.96	0.0002
Machine(Labor)	8.52E+08	6	1.42E+08	10.83	0
Scenario	907675	2	453838	0.03	0.966
Method*Labor	3.44E+08	6	5.73E+07	4.37	0.0003
Method*Machine(Labor)	1.19E+09	18	6.63E+07	5.06	0
Method*Scenario	1.68E+07	6	2.80E+06	0.21	0.9722
Labor*Scenario	3.78E+07	4	9.45E+06	0.72	0.5787
Scenario*Machine(Labor)	7.74E+07	12	6.45E+06	0.49	0.9181
Method*Labor*Scenario	6.02E+07	12	5.01E+06	0.38	0.9688
Method*Scenario*Machine(Labor)	1.54E+08	36	4.28E+06	0.33	0.9999
Residual	2.81E+09	214	1.31E+07		
Total (corrected)	6.19E+09	323			

Tukey's test indicates that P-LPR is more efficient in low than medium and high machine flexibilities when labor flexibility is medium. There was no significant difference between the efficiency of the methods in different machine flexibilities. Table 7.50 shows the summary result.

Table 7.50 Multiple comparison on machine flexibility levels within labor flexibility by fixing lower bounding methods

Method	Labor flexibility	Machine flexibility
B&B	High	-
	Medium	-
	Low	-
LBBD	High	-
	Medium	-
	Low	-
F-LPR	High	-
	Medium	-
	Low	-
P-LPR	High	-
	Medium	Low>>Medium=High
	Low	-

The analysis revealed that statistically, P-LPR is more efficient than B&B, LBBD and F-LPR in low machine flexibility and medium labor flexibility. Table 7.51 shows the summary result.

Table 7.51 Multiple comparison on lower bounding methods by fixing labor flexibility

Labor flexibility	Machine flexibility	Method
High	High	-
	Medium	-
	Low	-
Medium	High	-
	Medium	-
	Low	P-LPR>>F-LPR=LBBD=B&B
Low	High	-
	Medium	-
	Low	-

Table 7.52 shows that P-LPR is more efficient in medium than low and high labor flexibilities.

Table 7.52 Multiple comparison on labor flexibility by fixing lower bounding methods flexibility

Method	Labor flexibility
B&B	-
LBBD	-
F-LPR	-
P-LPR	Medium>>Low=High

7.2.3.3. Effectiveness of search algorithms

As stated before, in the absence of an optimal solution, the deviations are obtained from the best lower bounding method and unfortunately this deviation can mask the good quality of the upper bound. Table 7.53 shows the percentage deviation between the upper bound found from search algorithms and the B&B method, and the best lower bound. As shown in the table the quality of B&B, TS and TS-CL are almost the same and TS-IL is less effective than the other methods.

Table 7.53 Search algorithms deviation with best lower bound in large examples

Labor Flexibility	**Machine Flexibility**	**B&B**	**TS**	**TS-CL**	**TS-IL**	**Average**
High	**High**	47.93%	41.20%	41.30%	55.03%	**46.36%**
	Medium	30.05%	34.37%	34.37%	53.58%	**38.09%**
	Low	7.03%	5.60%	7.28%	9.97%	**7.47%**
	Average	**28.34%**	**27.06%**	**27.65%**	**39.53%**	**30.64%**
Medium	**High**	56.33%	58.88%	59.48%	70.09%	**61.19%**
	Medium	51.94%	51.22%	51.22%	80.31%	**58.67%**
	Low	34.35%	23.75%	23.75%	41.23%	**30.77%**
	Average	**47.54%**	**44.62%**	**44.82%**	**63.88%**	**50.21%**
Low	**High**	63.72%	68.43%	68.88%	74.20%	**68.81%**
	Medium	51.63%	60.62%	60.62%	69.51%	**60.59%**
	Low	26.89%	28.33%	28.33%	36.29%	**29.96%**
	Average	**47.42%**	**52.46%**	**52.61%**	**60.00%**	**53.12%**
Average		**41.10%**	**41.38%**	**41.69%**	**54.47%**	**44.66%**

As stated in medium size problems, the goal here is to show that the search algorithms have the same effectiveness as B&B method. Table 7.55 shows the average percentage deviation between the search algorithms and the best upper bound found. As shown in the table, B&B deviation is 2.5%, the tabu search algorithms all together were able to find solutions with only an average deviation of 2.82%. The TS-CL and TS-IL deviations were recorded as 3.08% and 12.34%, respectively.

Table 7.54 Search algorithms deviation with best upper bound in large examples

Labor Flexibility	Machine Flexibility	B&B	TS	TS-CL	TS-IL	Average
High	High	7.11%	2.18%	2.24%	12.55%	**6.02%**
	Medium	0.51%	4.01%	4.01%	19.16%	**6.92%**
	Low	3.00%	1.78%	3.36%	6.06%	**3.55%**
	Average	**3.54%**	**2.66%**	**3.20%**	**12.59%**	**5.50%**
Medium	High	0.67%	2.57%	2.96%	9.94%	**4.04%**
	Medium	1.93%	2.33%	2.33%	20.88%	**6.87%**
	Low	8.06%	1.22%	1.22%	15.03%	**6.38%**
	Average	**3.55%**	**2.04%**	**2.17%**	**15.28%**	**5.76%**
Low	High	0.01%	2.93%	3.25%	6.52%	**3.18%**
	Medium	0.20%	6.12%	6.12%	12.17%	**6.15%**
	Low	1.01%	2.23%	2.23%	8.75%	**3.55%**
	Average	**0.40%**	**3.76%**	**3.87%**	**9.15%**	**4.30%**
Average		**2.50%**	**2.82%**	**3.08%**	**12.34%**	**5.18%**

The experimental analysis showed that there is a significant difference between the quality of solutions found by different search algorithms. In addition there is a difference between the qualities of solutions in different machine flexibilities within labor flexibility. The interaction between search algorithms and labor flexibility and the interaction between search algorithms and machine flexibility within labor flexibility are also statistically significant. The results show that scenario and labor flexibility do not have any correlation with the quality of search algorithm. The ANOVA is shown in Table 7.55.

Table 7.55 ANOVA for search algorithms deviations in large examples

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Block	186397	2	93198.7	0.29	0.7505
Method	5.55E+07	3	1.85E+07	57	0
Labor	1.32E+06	2	658986	2.03	0.1336
Machine(Labor)	5.74E+06	6	956747	2.95	0.0087
Scenario	590550	2	295275	0.91	0.4039
Method*Labor	6.38E+06	6	1.06E+06	3.28	0.0042
Method*Machine(Labor)	1.59E+07	18	880792	2.72	0.0003
Method*Scenario	1.24E+06	6	205841	0.63	0.7024
Labor*Scenario	847018	4	211754	0.65	0.6254
Scenario*Machine(Labor)	7.16E+06	12	596882	1.84	0.0435
Method*Labor*Scenario	2.02E+06	12	168025	0.52	0.9018
Method*Scenario*Machine(Labor)	8.09E+06	36	224611	0.69	0.9058
Residual	6.94E+07	214	324303		
Total (corrected)	1.74E+08	323			

Pair-wise comparisons show that the effectiveness of B&B, TS and TS-CL does not change with machine flexibility but TS-IL has a better quality in low than medium machine flexibility when the labor flexibility is high and it performs better in high than medium machine flexibility with medium labor flexibility. Table 7.56 shows the summary result.

Table 7.56 Multiple comparison on machine flexibility levels within labor flexibility by fixing search algorithms

Method	Labor flexibility	Machine flexibility
B&B	High	-
	Medium	-
	Low	-
TS	High	-
	Medium	-
	Low	-
CL	High	-
	Medium	-
	Low	-
IL	High	Low>>Medium
	Medium	High>>Medium
	Low	-

Detail analysis shows that the quality of TS and TS-CL is the same as B&B, however they all have a better quality than TS-IL in medium machine flexibility and in high and medium labor flexibilities. When machine flexibility is low and labor flexibility is medium TS and TS-CL perform better than TS-IL. In medium machine and low labor flexibilities B&B has a better quality than TS-IL. Table 7.57 shows the summary result.

Table 7.57 Multiple comparison on search algorithms by fixing machine flexibility within labor flexibility

Labor flexibility	Machine flexibility	Method
High	High	-
	Medium	B&B=TS=CL>>IL
	Low	-
Medium	High	-
	Medium	B&B=TS=CL>>IL
	Low	TS=CL>>IL
Low	High	-
	Medium	B&B>>IL
	Low	-

7.2.3.4. Efficiency of search algorithms

Now that the statistical analysis showed that the effectiveness of the search algorithm TS-CL is as good as B&B, the investigation is extended to assess its efficiency. Table 7.58 is a summary table that shows the time spent by each search algorithm in seconds.

Table 7.58 Time spent by search algorithms in large examples

Labor Flexibility	**Machine Flexibility**	**B&B**	**TS**	**TS-CL**	**TS-IL**	**Average**
High	**High**	28800	16087	16087	1470	**15611**
	Medium	28800	15165	15165	2335	**15366**
	Low	28800	1086	1067	203	**7789**
	Average	**28800**	**10780**	**10773**	**1336**	**12922**
Medium	**High**	28800	16675	16675	3661	**16453**
	Medium	28800	14893	14893	688	**14819**
	Low	28800	10841	10841	1150	**12908**
	Average	**28800**	**14136**	**14136**	**1833**	**14726**
Low	**High**	28800	9974	9747	1257	**12444**
	Medium	28800	8560	8560	1219	**11785**
	Low	28800	9277	9277	1083	**12109**
	Average	**28800**	**9270**	**9195**	**1186**	**12113**
Average		**28800**	**11395**	**11368**	**1452**	**13254**

The result shows a huge difference between the efficiency of the search algorithms and the B&B. TS-IL is the most efficient algorithm that used on average 1452 s, next TS-CL which spent 11368 s, the combined TS used 11395 s and B&B spent 28800 s. In fact TS-IL is 7 times more efficient than TS-CL and 19 times more efficient than B&B. However, one can argue that the quality of solution found by TS-IL is not as good as B&B and TS-CL. TS-CL is also 2.5 times more efficient than B&B and it has the same quality as B&B.

Statistical analysis shows that there is a significant difference between the efficiency of different search algorithms. Also it states that labor flexibility and machine flexibility are affecting the efficiency. There is a significant difference between the efficiency of different scenarios. The interaction effects are not significant. Table 7.59 shows the ANOVA.

Table 7.59 ANOVA for search algorithms time in large examples

Source	Sum of Squares	Df	Mean Square	F-Ratio	P-Value
Block	2.39E+09	2	1.19E+09	26.79	0
Method	3.14E+10	3	1.05E+10	235.28	0
Labor	3.87E+08	2	1.93E+08	4.34	0.0142
Machine(Labor)	1.66E+09	6	2.76E+08	6.21	0
Scenario	5.17E+08	2	2.58E+08	5.8	0.0035
Method*Labor	2.98E+08	6	4.97E+07	1.12	0.3532
Method*Machine(Labor)	1.29E+09	18	7.19E+07	1.61	0.0584
Method*Scenario	3.28E+08	6	5.47E+07	1.23	0.2926
Labor*Scenario	5.86E+07	4	1.46E+07	0.33	0.8584
Scenario*Machine(Labor)	4.92E+08	12	4.10E+07	0.92	0.526
Method*Labor*Scenario	9.22E+07	12	7.68E+06	0.17	0.9992
Method*Scenario*Machine(Labor)	5.63E+08	36	1.56E+07	0.35	0.9998
Residual	9.53E+09	214	4.45E+07		
Total (corrected)	4.90E+10	323			

Tukey’s test indicates methods are more efficient in low than medium and high machine flexibilities and in high labor flexibility. The machine flexibility does not affect the efficiency in medium and low labor flexibilities. Table 7.60 shows the summary result.

Table 7.60 Multiple comparison on machine flexibility levels within labor flexibility

Labor flexibility	Machine flexibility
High	Low>>Medium=High
Medium	-
Low	-

Further analysis shows that TS-IL is the most efficient algorithm and TS and TS-CL have the same efficiency and they are all more efficient than B&B. Table 7.61 summarizes the results.

Table 7.61 Multiple comparison on search algorithms by fixing labor flexibility

Labor flexibility	Method
High	IL>>TS=CL>>B&B
Medium	IL>>TS=CL>>B&B
Low	IL>>TS=CL>>B&B

Table 7.62 shows that methods are more efficient in 30-70% scenario than 50-50%.

Table 7.62 Multiple comparison on scenario

Scenario
30-70%>>50-50%

The results from large structure show that TS-CL is as effective as B&B method and it is 2.5 times more efficient than B&B, which attests to the power of the developed search algorithms.

7.2.4. Application in Industry

Hybrid flowshop scheduling is the most generalized arrangement that can be reduced into other forms of scheduling and be used by different industries easily. Whether the industry has single machine, flowshop, identical-parallel machines, unrelated-parallel machines, flexible flowshop or hybrid flowshop arrangement, the findings from this research can be used. For example if all of the stages in the hybrid flowshop have single machine, a flowshop arrangement can be obtained. The same thing is true for other assumptions such as dual resources, sequence-dependent setup time, stage skipping and dynamic machine availability and job release times.

Industries are more interested in using search algorithms that are both effective and efficient. Lots of different results were proposed here for the effectiveness and efficiency of the algorithms in different structures, labor and machine flexibilities. In order for an industry to find out which algorithm to choose, first they should figure out what size of problems they are using by looking at the total number of jobs, stages and skills. After the size of the problem, small, medium or large, is decided, labor and machine flexibilities should be identified (based on data generation section). Then the appropriate statistical analysis table should be used. For instance, if Company A has a medium size problem with high labor and low machine flexibilities, Table 7.38 should be used for finding the most effective algorithms. In high labor flexibilities there is no significant difference between the quality of algorithms, however the average deviation for TS-CL is 3.15% whereas the average deviation for TS-IL is 4.03%. We can see that TS-CL has a better quality than TS-IL but this difference is not statistically significant and can be due to random errors (noises). Then the second table to look at is Table 7.42, which helps us find the most efficient algorithm. There is no difference between the efficiency of search algorithms as well. TS-CL used in average 1263 s and TS-IL used 348 s. We can see a difference but as stated in effectiveness, this difference is not statistically significant. So the conclusion here is that statistically there is no significant difference between TS-CL and TS-IL effectiveness and efficiency, however TS-CL is showing a better quality and TS-IL, on average, is faster. The final decision is a judgment call and it depends on the goals and vision of the industry.

Now consider another example. Company B, has a large size problem and they have medium labor and low machine flexibilities. Table 7.57 helps to choose the most effective algorithm. Based on this table TS-CL is more effective than TS-IL and based on Table 7.61 TS-IL is more efficient than TS-CL. The average deviation for TS-CL and

TS-IL are 1.22% and 15.03%. However, TS-CL and TS-IL used 10831 s and 1150 s, respectively. If company B is quality driven and it can forgo the time difference, TS-CL is the best choice. If Company B is concerned about investing too much time and they can forgo of the quality difference, TS-IL is the best choice.

7.2.5. Summary

In this chapter statistical analysis was used to assess the effectiveness and efficiency of lower bounding methods and search algorithms.

Small size problems: B&B method was capable to solve all examples optimally. Lower bounding methods all together were capable of providing lower bounds that are statistically the same as optimal solutions. There was no significant difference between the efficiency of B&B, F-LPR and P-LPR, however LBBD was less efficient than the other methods in some labor and machine flexibilities. There was no significant difference between the TS-TCL and TS-CL and the optimal solution, but TS-IL was less effective than the rest of the search algorithms in some labor and machine flexibilities. All search algorithms showed the same efficiency, but they were more efficient than B&B in some labor and machine flexibilities.

Medium size problems: Only 20.99% of the problems were solved optimally. P-LPR was the most effective lower bounding method and was capable to provide a better lower bound than B&B. LBBD was the least effective lower bounding method, however there was no significant difference between the quality of its lower bound and the lower bound found by B&B. The results also showed that P-LPR is the most efficient method. There was no significant difference between the quality of TS-CL and B&B, however TS-IL was less effective. Both TS-CL and TS-IL were more efficient than B&B.

Large size problems: None of the examples were solved optimally in this category. LBBD was the most effective lower bounding method, and almost all of the lower bounding methods including B&B have the same efficiency. There was no significant difference between the effectiveness of TS-CL and B&B, but TS-IL had a lower quality in some labor and machine flexibilities. TS-IL was more efficient than TS-CL and they were both more efficient than B&B.

8. CONCLUSIONS

This dissertation addressed a hybrid-flow shop scheduling problem in the presence of dual resource constraints. The dual resources were machine and labor. Different skill levels were considered for labor and the skill level required to perform the setup could be different from that needed to perform the run. The setup time was sequence-dependent, and job release times and machine availability times were dynamic. Also machine skipping was allowed. The problem was shown to be NP-hard in the strong sense with the bicriteria objective function of minimizing the weighted sum of weighted completion time and weighted tardiness.

A linear mixed-integer mathematical programming model was proposed and CPLEX solver was used to find solutions with branch-and-bound (B&B) technique. Also three different meta-search heuristic algorithms based on tabu search were developed in order to quickly solve the scheduling problems. The tabu search based heuristics were designed to handle the dual resource scheduling problem at two layers merged together, where each layer was a tabu search algorithm itself. The outside layer was designed to find schedule of jobs on machine and the inside layer was designed to find schedule of jobs on labor. TS-TCL was developed by applying the labor layer to machine schedule found by every TCL values, TS-CL was developed by applying the labor layer to machine schedule found by every CL values and finally TS-IL was developed by applying the labor layer to machine schedule found by every IL values. In addition, three different lower bounding methods were proposed. The first method was logic-based Benders decomposition (LBBD) and the second and third methods were two different variations of iterative selective LP relaxation called fractional LP relaxation (F-LPR) and positive LP relaxation (P-LPR).

A total of 243 examples were generated in small, medium and large size problems. An experimental analysis based on a nested-factorial design with blocking was developed in order to identify statistically significant differences between the effectiveness and efficiency of the lower bounding methods and search algorithms. In small size problems all of the examples were solved optimally by B&B. The results from statistical analysis showed that there was a significant difference between effectiveness of different lower bounding methods, and labor flexibilities and machine flexibilities were affecting the quality of lower bound. The average deviation from optimal solution in B&B, LBBD, F-LPR and P-LPR were 0%, 1.34%, 1.34% and 1.41% respectively. In high labor flexibilities, there was no difference between the effectiveness of lower bounding methods and optimal solutions found by B&B. In medium labor there was no difference between the quality of lower bound found from LBBB and the optimal solution but both F-LPR and P-LPR were less effective than the optimal solution. In low labor flexibility both F-LPR and P-LPR were as effective as the optimal solution but LBBD was less effective. There was a significant difference between the efficiency of different lower bounding methods, and both labor flexibility and machine flexibility were affecting the efficiency. On average B&B, LBBD, F-LPR and P-LPR took 373, 6119, 79 and 84 s, respectively to find their lower bound. In labor and machine flexibilities of medium-high, low-high and low-medium B&B was as efficient as both F-LPR and P-LPR and LBBD had the least efficiency.

There was a significant difference between the quality of different search algorithms, and both labor flexibility and machine flexibility were affecting the effectiveness of upper bounds. The average deviations of B&B, TS-TCL, TS-CL and TS-IL from optimal solutions were 0%, 1.03%, 1.5% and 4.5%, respectively. There was no significant difference between the quality of TS-TCL and TS-CL and the optimal solution. In low labor and high and medium machine flexibilities TS-IL was less effective than the rest of the search algorithms. There was a significant difference between the efficiency of different search algorithms, and both labor flexibility and machine flexibility were affecting the efficiency. On average B&B, TS-TCL, TS-CL and TS-IL took 373, 265, 26 and 5 s, respectively to find their solution. Further analysis showed that all search algorithms had the same efficiency but in medium labor and high machine flexibilities B&B was less efficient than the search algorithms.

In medium size problems only 20.99% of the examples were solved optimally. B&B solved 19.75% of the examples optimally whereas LBBD, F-LPR and P-LPR solved 13.58%, 17.28% and 12.35% of the examples optimally. For the rest of the examples only a good lower bound was found. The results from statistical analysis showed that lower bounding methods, labor flexibilities and machine flexibilities affect the quality of lower bounds. The average deviation from the best lower bound in B&B, LBBD, F-LPR and P-LPR were 9.7%, 5.73%, 3.86% and 0.95% respectively. Lower bounding methods showed the same effectiveness in most of the labor and machine flexibilities, but P-LPR performed better than B&B in medium labor and high machine flexibilities. In low labor flexibility, P-LPR performed better than both B&B and LBBD and F-LPR performed better than B&B in high machine flexibilities. Also P-LPR had a better quality than B&B in medium machine flexibility. Both P-LPR and F-LPR outperformed LBBD in low machine flexibility. There was a significant difference between the efficiency of different lower bounding methods, and both labor flexibility and machine flexibility were affecting the efficiency. On average B&B, LBBD, F-LPR and P-LPR took 23350, 25089, 18515 and 14136 s, respectively to find their lower bound. Analysis revealed that statistically, P-LPR was more efficient than B&B and LBBD in high and medium labor flexibility.

There was a significant difference between the quality of different search algorithms, and labor flexibility was affecting the effectiveness of upper bounds. The average deviations of B&B, TS, TS-CL and TS-IL from optimal solutions were 0.49%, 3.05% and 9.35%, respectively. The quality of TS-CL was the same as B&B. However, they both had a better quality than TS-IL in medium and low labor flexibilities. There was a significant difference between the efficiency of different search algorithms, and both labor flexibility and machine flexibility were affecting the efficiency. On average B&B, TS-TCL, TS-IL took 23661, 2295, and 529 s, respectively to find their solution. Further analysis showed that there is no statistical difference between TS-CL and TS-IL efficiency and they were both more efficient than B&B in different levels of machine flexibility within labor flexibility, except that in low machine flexibility and high labor flexibility there was no significant difference between search algorithms and B&B.

In large size problems none of the examples were solved optimally and only a lower bound was found. The results from statistical analysis showed that lower bounding methods and machine flexibilities affect the quality of lower bounds. The average deviation from the best lower bound in B&B, LBBD, F-LPR and P-LPR were 26.32%, 2.51%, 9.44% and 12.49% respectively. LBBD performed better than B&B and P-LPR in both high and medium machine flexibilities and LBBD, F-LPR and P-LPR had the same quality and they all had a better quality than B&B in low machine flexibility when labor flexibility is high. LBBD and F-LPR outperformed B&B in high machine flexibility and LBBD’s quality was better than B&B in medium and low machine flexibilities when labor flexibility is medium. Also LBBD, F-LPR and P-LPR performed the same and they all had a better quality than B&B in medium machine flexibility and LBBD had a better quality than B&B in low machine flexibility when labor flexibility was low. There was a significant difference between the efficiency of different lower bounding methods, and both labor flexibility and machine flexibility were affecting the efficiency. On average B&B, LBBD, F-LPR and P-LPR took 28800, 28800, 27881 and 26647 s, respectively to find their lower bound. The analysis revealed that statistically, P-LPR was more efficient than B&B, LBBD and F-LPR in low machine flexibility and medium labor flexibility.

There was a significant difference between the quality of different search algorithms, and machine flexibility was affecting the effectiveness of upper bounds. The average deviations of B&B, TS, TS-CL and TS-IL from optimal solutions were 2.5%, 3.08% and 12.34%, respectively. Detail analysis showed that the quality of TS-CL was the same as B&B, however they both had a better quality than TS-IL in medium machine flexibility and in high and medium labor flexibilities. When machine flexibility was low and labor flexibility was medium TS-CL performed better than TS-IL. In medium machine and low labor flexibilities B&B had a better quality than TS-IL. There was a significant difference between the efficiency of different search algorithms, also labor flexibility, machine flexibility and scenarios were affecting the efficiency. On average B&B, TS-TCL, TS-IL took 28800, 11368, and 1452 s, respectively to find their solution. Further analysis showed that TS-IL was more efficient than TS-CL and they were both more efficient than B&B.

As for future research, the search algorithms can be improved. Although the TS-IL is very efficient it is not as effective as TS-CL. TS-CL can lose its efficiency in larger size problems. A combination of tabu search with other heuristic algorithms, other than the genetic algorithm which has already been investigated in this research, can be helpful.

The LBBD method’s performance improved as the size of the problems increased. In small size problems LBBD showed a good performance but it wasn’t effective and efficient in some of the instances at all. In medium size problems, it finds better lower bounds than B&B but it was not as effective and efficient as LP-relaxations. In large problems, however, LBBD showed an outstanding performance but no comment can be made about the efficiency because most of the methods were forced to terminate after 8 hours. The reason that the quality of LBBD seems to improve is because of the decrease in quality of the rest of the methods (B&B and LP relaxations) when the size of the problem increases. Note that the quality of LBBD is also decreasing, but the deviation for a lower bound is obtained from the best lower bound found from all of the methods. As the size of the problem grows the best lower bound is mostly coming from LBBD method, therefore it seems like that the quality of LBBD is not changing. In fact the loss of quality in LBBD is happening more slowly than B&B and LP relaxations. The reason that LBBD is not showing a great performance with a high efficiency in small size problems is that when using LBBD, master problem and subproblems that are generated should not be NP-hard, so LBBD can perform lots of iterations between master and subproblems and find the optimal solution. In our decomposition both master and subproblems are NP-hard and each iteration is very time consuming. Using another decomposition technique is recommended for obtaining even better lower bounds than LBBD.

In addition, the quality of the LP relaxation methods is highly related to their variable selecting decisions. In every iteration a selective group of variables is selected and forced to be binary for the next generation. Revising the decision rules can improve the effectiveness and efficiency.

BIBLIOGRAPHY

[1] Gargeya, V. B., & Deane, R. H. (1996). Scheduling research in multiple resource constrained job shops: a review and critique. *International Journal of Production Research*, *34*(8), 2077-2097.

[2] Johnson, S. M. (1954). Optimal two-and three-stage production schedules with setup times included. *Naval research logistics quarterly*, *1*(1), 61-68.

[3] Linn, R., & Zhang, W. (1999). Hybrid flow shop scheduling: a survey. *Computers & Industrial Engineering*, *37*(1), 57-61.

[4] Ghirardi, M., & Potts, C. N. (2005). Makespan minimization for scheduling unrelated parallel machines: A recovering beam search approach. *European Journal of Operational Research*, *165*(2), 457-467.

[5] Ribas, I., Leisten, R., & Framiñan, J. M. (2010). Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. *Computers & Operations Research, 37*(8), 1439-1454.

[6] Chen, Z. L., & Vairaktarakis, G. L. (2005). Integrated scheduling of production and distribution operations. *Management Science*, *51*(4), 614-628.

[7] Eren, T., & Güner, E. (2006). A bicriteria scheduling with sequence-dependent setup times. *Applied Mathematics and Computation*, *179*(1), 378-385.

[8] Mansouri, S. A., Hendizadeh, S. H., & Salmasi, N. (2009). Bicriteria scheduling of a two-machine flowshop with sequence-dependent setup times. *The International Journal of Advanced Manufacturing Technology*, *40*(11-12), 1216-1226.

[9] Köksalan, M., & Burak Keha, A. (2003). Using genetic algorithms for single-machine bicriteria scheduling problems. *European Journal of Operational Research*, *145*(3), 543-556.

[10] Chauhan, S. S., Gordon, V., & Proth, J. M. (2007). Scheduling in supply chain environment. *European Journal of Operational Research*, *183*(3), 961-970.

[11] Chen, Z. L., & Hall, N. G. (2007). Supply chain scheduling: Conflict and cooperation in assembly systems. *Operations Research*, *55*(6), 1072-1089.

[12] Chou, F. D., & Lee, C. E. (1999). Two-machine flowshop scheduling with bicriteria problem. *Computers & industrial engineering*, *36*(3), 549-564.

[13] Nelson, R. T. (1970). A simulation of labor efficiency and centralized assignment in a production model. *Management Science, 17*(2), B-97.

[14] Fryer, J. S. (1973). Operating policies in multiechelon dual-constraint job shops. *Management Science*, *19*(9), 1001-1012.

[15] Weeks, J. K., & Fryer, J. S. (1977). A methodology for assigning minimum cost due-dates. *Management Science*, *23*(8), 872-881.

[16] Weeks, J. K. (1979). A simulation study of predictable due-dates. *Management Science*, *25*(4), 363-373.

[17] Gunther, R.E. (1981). Dual response parallel queues with server transfer and information access delays. *Decision Science*, 12, 197–211.

[18] Treleven, M. D., & Elvers, D. A. (1985). An investigation of labor assignment rules in a dual-constrained job shop. *Journal of Operations Management*, *6*(1), 51-68.

[19] Avi-Itzhak, B., Maxwell, W. L., & Miller, L. W. (1965). Queuing with alternating priorities. *Operations Research*, *13*(2), 306-318.

[20] Nelson, R. T. (1966). Labor assignment as a dynamic control problem. *Operations Research*, *14*(3), 369-376.

[21] Takács, L. (1968). Two queues attended by a single server. *Operations Research*, *16*(3), 639-650.

[22] Xu, J., Xu, X., & Xie, S. Q. (2011). Recent developments in Dual Resource Constrained (DRC) system research. *European Journal of Operational Research*, *215*(2), 309-318.

[23] Lobo, B. J., Hodgson, T. J., King, R. E., Thoney, K. A., & Wilson, J. R. (2013). An effective lower bound on Lmax in a worker-constrained job shop. *Computers & Operations Research*, *40(1),* 328–343.

[24] Lobo, B. J., Hodgson, T. J., King, R. E., Thoney, K. A., & Wilson, J. R. (2013). Allocating job-shop manpower to minimize Lmax: Optimality criteria, search heuristics, and probabilistic quality metrics. *Computers & Operations Research*. http://dx.doi.org/10.1016/j.cor.2013.02.008.

[25] ElMaraghy, H., Patel, V., & Abdallah, I. B. (1999). A Genetic Algorithm Based Approach for Scheduling of Dual-Resource Constrainded Manufacturing Systems. *CIRP Annals-Manufacturing Technology*, *48*(1), 369-372.

[26] ElMaraghy, H., Patel, V., & Abdallah, I. B. (2000). Scheduling of manufacturing systems under dual-resource constraints using genetic algorithms. *Journal of manufacturing Systems*, *19*(3), 186-201.

[27] Chaudhry, I. A., & Drake, P. R. (2009). Minimizing total tardiness for the machine scheduling and worker assignment problems in identical parallel machines using genetic algorithms. *The International Journal of Advanced Manufacturing Technology*, *42*(5-6), 581-594.

[28] Mehravaran, Y., & Logendran, R. (2013). Non-permutation flowshop scheduling with dual resources. *Expert Systems with Applications*. http://dx.doi.org/10.1016/j.eswa.2013.03.007.

[29] Suresh, V., & Chaudhuri, D. (1996). Bicriteria scheduling problem for unrelated parallel machines. *Computers & industrial engineering*, *30*(1), 77-82.

[30] Logendran, R., & Subur, F. (2004). Unrelated parallel machine scheduling with job splitting. *IIE Transactions*, *36*(4), 359-372.

[31] Logendran, R., McDonell, B., & Smucker, B. (2007). Scheduling unrelated parallel machines with sequence-dependent setups. *Computers & Operations Research*, *34*(11), 3420-3438.

[32] Mehravaran, Y., & Logendran, R. (2011). Bicriteria supply chain scheduling on unrelated-parallel machines. *Journal of the Chinese Institute of Industrial Engineers*, *28*(2), 91-101.

[33] Yaghubian, A. R., Hodgson, T. J., Joines, J. A., Culbreth, C. T., & Huang, J. C. (1999). Dry kiln scheduling in furniture production. *IIE transactions*, *31*(8), 733-738.

[34] Azizoglu, M. E. R. A. L., & Kirca, O. M. E. R. (1999). Scheduling jobs on unrelated parallel machines to minimize regular total cost functions. *IIE transactions*, *31*(2), 153-159.

[35] Ying, K. C. (2008). Solving non-permutation flowshop scheduling problems by an effective iterated greedy heuristic. *The International Journal of Advanced Manufacturing Technology*, *38*(3-4), 348-354.

[36] Tavakkoli-Moghaddam, R., Taheri, F., Bazzazi, M., Izadi, M., & Sassani, F. (2009). Design of a genetic algorithm for bi-objective unrelated parallel machines scheduling with sequence-dependent setup times and precedence constraints. *Computers & Operations Research*, *36*(12), 3224-3230.

[37] Fanjul-Peyro, L., & Ruiz, R. (2010). Iterated greedy local search methods for unrelated parallel machine scheduling. *European Journal of Operational Research*, *207*(1), 55-69.

[38] Rocha, P. L., Ravetti, M. G., Mateus, G. R., & Pardalos, P. M. (2008). Exact algorithms for a scheduling problem with unrelated parallel machines and sequence and machine-dependent setup times. *Computers & Operations Research*, *35*(4), 1250-1264.

[39] Onwubolu, G. C., & Mutingi, M. (1999). Genetic algorithm for minimizing tardiness in flow-shop scheduling. *Production planning & control*, *10*(5), 462-471.

[40] Ruiz, R., & Maroto, C. (2005). A comprehensive review and evaluation of permutation flowshop heuristics. *European Journal of Operational Research*,*165*(2), 479-494.

[41] Srikar, B. N., & Ghosh, S. (1986). A MILP model for the n-job, m-stage flowshop with sequence dependent set-up times. *International Journal of Production Research*, *24*(6), 1459-1474.

[42] Liao, L. M., & Huang, C. J. (2010). Tabu search for non-permutation flowshop scheduling problem with minimizing total tardiness. *Applied Mathematics and Computation*, *217*(2), 557-567.

[43] Aggoune, R., & Portmann, M. C. (2006). Flow shop scheduling problem with limited machine availability: a heuristic approach. *International Journal of Production Economics*, *99*(1), 4-15.

[44] Mehravaran, Y., & Logendran, R. (2012). Non-permutation flowshop scheduling in a supply chain with sequence-dependent setup times. *International Journal of Production Economics*, *135*(2), 953-963.

[45] Elmaghraby, S.E., & Kamoub, R.E. (1997). Production control in hybrid flowshops: an example from textile manufacturing. In: *The Planning and Scheduling of Production Systems* (Artiba, A., & Elmaghraby, S.E. ed.). Chap. 6, Chapman & Hall, UK.

[46] Jin, Z., Yang, Z., & Ito, T. (2006). Metaheuristic algorithms for the multistage hybrid flowshop scheduling problem. *International Journal of Production Economics*, *100*(2), 322-334.

[47] Tang, L., Xuan, H., & Liu, J. (2006). A new Lagrangian relaxation algorithm for hybrid flowshop scheduling to minimize total weighted completion time. *Computers & Operations Research*, *33*(11), 3344-3359.

[48] Portmann, M. C., Vignier, A., Dardilhac, D., & Dezalay, D. (1998). Branch and bound crossed with GA to solve hybrid flowshops. *European Journal of Operational Research*, *107*(2), 389-400.

[49] Brah, S. A., & Hunsucker, J. L. (1991). Branch and bound algorithm for the flow shop with multiple processors. *European Journal of Operational Research*, *51*(1), 88-99.

[50] Lee, G. C., & Kim, Y. D. (2004). A branch-and-bound algorithm for a two-stage hybrid flowshop scheduling problem minimizing total tardiness. *International Journal of Production Research*, *42*(22), 4731-4743.

[51] Haouari, M., & M'Hallah, R. (1997). Heuristic algorithms for the two-stage hybrid flowshop problem. *Operations Research Letters*, *21*(1), 43-53.

[52] Alaykýran, K., Engin, O., & Döyen, A. (2007). Using ant colony optimization to solve hybrid flow shop scheduling problems. *The international journal of advanced manufacturing technology*, *35*(5-6), 541-550.

[53] Allaoui, H., & Artiba, A. (2006). Scheduling two-stage hybrid flow shop with availability constraints. *Computers & Operations Research*, *33*(5), 1399-1419.

[54] Zandieh, M., Fatemi Ghomi, S. M. T., & Moattar Husseini, S. M. (2006). An immune algorithm approach to hybrid flow shops scheduling with sequence-dependent setup times. *Applied Mathematics and Computation*, *180*(1), 111-127.

[55] Choi, S. W., Kim, Y. D., & Lee, G. C. (2005). Minimizing total tardiness of orders with reentrant lots in a hybrid flowshop. *International Journal of Production Research*, *43*(11), 2149-2167.

[56] Gupta, J. N. D., & Tunc, E. A. (1998). Minimizing tardy jobs in a two-stage hybrid flowshop. *International Journal of Production Research*, *36*(9), 2397-2417.

[57] Haouari, M., & Hidri, L. (2008). On the hybrid flowshop scheduling problem. *International Journal of Production Economics*, *113*(1), 495-497.

[58] Janiak, A., Kozan, E., Lichtenstein, M., & Oğuz, C. (2007). Metaheuristic approaches to the hybrid flow shop scheduling problem with a cost-related criterion. *International journal of production economics*, *105*(2), 407-424.

[59] Moursli, O., & Pochet, Y. (2000). A branch-and-bound algorithm for the hybrid flowshop. *International Journal of Production Economics*, *64*(1), 113-125.

[60] Naderi, B., Zandieh, M., & Roshanaei, V. (2009). Scheduling hybrid flowshops with sequence dependent setup times to minimize makespan and maximum tardiness. *The International Journal of Advanced Manufacturing Technology*,*41*(11-12), 1186-1198.

[61] Nishi, T., Hiranaka, Y., & Inuiguchi, M. (2010). Lagrangian relaxation with cut generation for hybrid flowshop scheduling problems to minimize the total weighted tardiness. *Computers & Operations Research*, *37*(1), 189-198.

[62] Riane, F., Artiba, A., & E Elmaghraby, S. (1998). A hybrid three-stage flowshop problem: efficient heuristics to minimize makespan. *European Journal of Operational Research*, *109*(2), 321-329.

[63] Tseng, C. T., & Liao, C. J. (2008). A particle swarm optimization algorithm for hybrid flow-shop scheduling with multiprocessor tasks. *International Journal of Production Research*, *46*(17), 4655-4670.

[64] Wang, X., & Tang, L. (2009). A tabu search heuristic for the hybrid flowshop scheduling with finite intermediate buffers. *Computers & Operations Research*, *36*(3), 907-918.

[65] Ying, K. C. (2008). An iterated greedy heuristic for multistage hybrid flowshop scheduling problems with multiprocessor tasks. *Journal of the Operational Research Society*, *60*(6), 810-817.

[66] Ying, K. C., & Lin, S. W. (2006). Multiprocessor task scheduling in multistage hybrid flow-shops: an ant colony system approach. *International Journal of Production Research*, *44*(16), 3161-3177.

[67] Ruiz, R., & Maroto, C. (2006). A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility. *European Journal of Operational Research*, *169*(3), 781-800.

[68] Bertel, S., & Billaut, J. C. (2004). A genetic algorithm for an industrial multiprocessor flow shop scheduling problem with recirculation. *European Journal of Operational Research*, *159*(3), 651-662.

[69] Low, C., Hsu, C. J., & Su, C. T. (2008). A two-stage hybrid flowshop scheduling problem with a function constraint and unrelated alternative machines.*Computers & Operations Research*, *35*(3), 845-853.

[70] Yaurima, V., Burtseva, L., & Tchernykh, A. (2009). Hybrid flowshop with unrelated machines, sequence-dependent setup time, availability constraints and limited buffers. *Computers & Industrial Engineering*, *56*(4), 1452-1463.

[71] Armentano, V. A., & Ronconi, D. P. (1999). Tabu search for total tardiness minimization in flowshop scheduling problems. *Computers & operations research*, *26*(3), 219-235.

[72] Silva, J. D. L., Burke, E. K., & Petrovic, S. (2004). An introduction to multiobjective metaheuristics for scheduling and timetabling. In *Metaheuristics for multiobjective optimisation* (pp. 91-129). Springer Berlin Heidelberg.

[73] Garey, M.R., & Johnson DS., editors (1979). *Computers and intractability: A guide to the theory of NP-completeness*, W.H. Freeman, New York.

[74] Garey, M. R., Johnson, D. S., & Sethi, R. (1976). The complexity of flowshop and jobshop scheduling. *Mathematics of operations research*, *1*(2), 117-129.

[75] Du, J., & Leung, J. Y. T. (1990). Minimizing total tardiness on one machine is NP-hard. *Mathematics of operations research*, *15*(3), 483-495.

[76] Lenstra, J. K., Shmoys, D. B., & Tardos, É. (1990). Approximation algorithms for scheduling unrelated parallel machines. *Mathematical programming*, *46*(1-3), 259-271.

[77] Pardalos, P. M., Pitsoulis, L., Mavridou, T., & Resende, M. G. (1995). Parallel search for combinatorial optimization: genetic algorithms, simulated annealing, tabu search and GRASP. In *Parallel Algorithms for Irregularly Structured Problems* (pp. 317-331). Springer Berlin Heidelberg.

[78] Jans, R., & Degraeve, Z. (2007). Meta-heuristics for dynamic lot sizing: a review and comparison of solution approaches. *European Journal of Operational Research*, *177*(3), 1855-1875.

[79] Glover, F. (2002). *Tabu Search*. Manuel Laguna. 6th edition, Boston.

[80] Vallada, E., Ruiz, R., & Minella, G. (2008). Minimising total tardiness in the m-machine flowshop problem: A review and evaluation of heuristics and metaheuristics. *Computers & Operations Research*, *35*(4), 1350-1373

[81] Murata, T., Ishibuchi, H., & Tanaka, H. (1996). Genetic algorithms for flowshop scheduling problems. *Computers & Industrial Engineering*, *30*(4), 1061-1071.

[82] Mantawy, A. H., Abdel-Magid, Y. L., & Selim, S. Z. (1999). Integrating genetic algorithms, tabu search, and simulated annealing for the unit commitment problem. *Power Systems, IEEE Transactions on*, *14*(3), 829-836.

[83] Glover, F. (1989). Tabu search—part I. *ORSA Journal on computing*, *1*(3), 190-206.

[84] Glover, F. (1990). Tabu search—part II. *ORSA Journal on computing*, *2*(1), 4-32.

[85] Glover, F. (1990). Tabu search: A tutorial. *Interfaces*, *20*(4), 74-94.

[86] Glover, F., & Kochenberger, G.A. (2003). *Handbook of metaheuristics*, illustrated ed. Kluwer Academic Publisher, London.

[87] Glover, F., & Laguna, M. (1997). *Tabu Search*, sixth ed. Kluwer Academic Publishers, London.

[88] Bilge, Ü., Kıraç, F., Kurtulan, M., & Pekgün, P. (2004). A tabu search algorithm for parallel machine total tardiness problem. *Computers & Operations Research*, *31*(3), 397-414.

[89] Choobineh, F. F., Mohebbi, E., & Khoo, H. (2006). A multi-objective tabu search for a single-machine scheduling problem with sequence-dependent setup times. *European Journal of Operational Research*, *175*(1), 318-337.

[90] Barnes, J.W., Laguna, M., & Glover. F. (1995). An overview of tabu search approaches to production scheduling problems, in Brown DE., Scherer, W.T., editors, *Intelligent Scheduling Systems*. Kluwer Academic Publishers, Boston, MA, 101-127.

[91] ILOG CPLEX Optimization Studio. (2009). IBM, Version 12.2.

[92] Hooker, J.N., & Yan, H. (1995). Logic circuit verification by Benders decomposition. in Saraswat, V., & Van Hentenryck, P., eds. *Principles and Practice of Constraint Programming: The Newport Papers*. MIT Press (Cambridge, MA) 267-288.

[93] Hooker, JN. (2000). *Logic-based Methods for Optimization: Combining Optimization and Constraint Satisfaction*. John Wiley & Sons.

[94] Hooker, J. N., & Ottosson, G. (2003). Logic-based Benders decomposition.*Mathematical Programming*, *96*(1), 33-60.

[95] Jain, V., & Grossmann, I. E. (2001). Algorithms for hybrid MILP/CP models for a class of optimization problems. *INFORMS Journal on computing*, *13*(4), 258-276.

[96] Hooker, J.N., & Ottosson, G. (2000). Logic-based Benders decomposition. *Tepper School of Business*. Paper 172. http://repository.cmu.edu/tepper/172.

[97] Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik*, *4*(1), 238-252.

[98] Geoffrion, A. M. (1972). Generalized benders decomposition. *Journal of optimization theory and applications*, *10*(4), 237-260.

[99] Hooker, J.N. (2004). A hybrid method for planning and scheduling. *Tepper School of Business*. Paper 164. http://repository.cmu.edu/tepper/164.

[100] Hooker, J.N. (2007). Planning and scheduling by logic-based Benders decomposition. *Tepper School of Business*. Paper 51. http://repository.cmu.edu/tepper/151.

[101] Chu, Y., & Xia, Q. (2005). A hybrid algorithm for a class of resource constrained scheduling problems. In *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems* (pp. 110-124). Springer Berlin Heidelberg.

[102] Hooker, J. N. (2005). Planning and scheduling to minimize tardiness. In *Principles and Practice of Constraint Programming-CP 2005* (pp. 314-327). Springer Berlin Heidelberg.

[103] Visual Studio C#. (2008).

[104] Montgomery, D.C. (2009). *Design and analysis of experiments*. 7th ed. New York: Wiley.

[105] Stat Point Technologies INC. (2010). STATGRAPHICS Centurion XVI: Version 16.1.02.

APPENDICES

APPENDIX A. Comparison between Tabu Search, Genetic Algorithm and Hybrid of Tabu Search-Genetic Algorithm

Table A.1 Effectiveness and efficiency of different search algorithm

Example		1	2	3	4	5	6
B&B	Optimal	465	349.3	1475.4	997.6	1633.8	1983.1*
	time	1	2	936	5	23395	28800
TS	Solution	465	363	1518.6	1025	1689.1	2095.1
	time	42	602	858	1137	3765	4130
GA	Solution	947.4	364.7	1509	1948.6	1923.4	2681.7
	time	8	3	13	20	64	71
Hybrid	Solution	945.7	389.9	1534	1739.4	1736.7	2557.1
	time	180	64	1500	4080	4800	3400

* Lower bound

APPENDIX B. Detailed Results from Small Size Problems

Table B.1 Branch-and-bound results in small examples

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	B&B		
					LB	UB	Time
Small	1	High	High	30-70%	674	674	4
Small	1	High	High	50-50%	1011	1011	1
Small	1	High	High	70-30%	1285.3	1285.3	1
Small	1	High	Low	30-70%	509	509	0
Small	1	High	Low	50-50%	422.5	422.5	0
Small	1	High	Low	70-30%	548.6	548.6	0
Small	1	High	Medium	30-70%	607.7	607.7	0
Small	1	High	Medium	50-50%	593.5	593.5	1
Small	1	High	Medium	70-30%	634.8	634.8	0
Small	1	Low	High	30-70%	611.3	611.3	1
Small	1	Low	High	50-50%	459	459	1
Small	1	Low	High	70-30%	713.5	713.5	2
Small	1	Low	Low	30-70%	938.9	938.9	2
Small	1	Low	Low	50-50%	1050.5	1050.5	0
Small	1	Low	Low	70-30%	1071.7	1071.7	0
Small	1	Low	Medium	30-70%	322.1	322.1	2
Small	1	Low	Medium	50-50%	479.5	479.5	2
Small	1	Low	Medium	70-30%	605.5	605.5	2
Small	1	Medium	High	30-70%	283.2	283.3	3
Small	1	Medium	High	50-50%	500.5	500.5	4
Small	1	Medium	High	70-30%	681	681	6
Small	1	Medium	Low	30-70%	418.2	418.2	2
Small	1	Medium	Low	50-50%	609	609	3
Small	1	Medium	Low	70-30%	657.6	657.6	2
Small	1	Medium	Medium	30-70%	306.7	306.7	36
Small	1	Medium	Medium	50-50%	409	409	24
Small	1	Medium	Medium	70-30%	575.6	575.6	15
Small	2	High	High	30-70%	855.7	855.7	85
Small	2	High	High	50-50%	1000	1000	16
Small	2	High	High	70-30%	1096.6	1096.6	31
Small	2	High	Low	30-70%	1266.5	1266.5	0
Small	2	High	Low	50-50%	1406.5	1406.5	0
Small	2	High	Low	70-30%	1479.4	1479.4	0
Small	2	High	Medium	30-70%	638.4	638.4	321
Small	2	High	Medium	50-50%	979.5	979.5	76
Small	2	High	Medium	70-30%	1421.1	1421.1	17
Small	2	Low	High	30-70%	462.9	462.9	19
Small	2	Low	High	50-50%	674	674	5
Small	2	Low	High	70-30%	954.6	954.6	6
Small	2	Low	Low	30-70%	1698.9	1698.9	8
Small	2	Low	Low	50-50%	1742	1742	3

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	B&B		
					LB	UB	Time
Small	2	Low	Low	70-30%	2184.5	2184.5	12
Small	2	Low	Medium	30-70%	1610.1	1610.1	97
Small	2	Low	Medium	50-50%	1820.5	1820.5	239
Small	2	Low	Medium	70-30%	2130.8	2130.8	184
Small	2	Medium	High	30-70%	570.3	570.3	4547
Small	2	Medium	High	50-50%	915	915	1904
Small	2	Medium	High	70-30%	1193.4	1193.4	1386
Small	2	Medium	Low	30-70%	890.1	890.1	6
Small	2	Medium	Low	50-50%	913.5	913.5	9
Small	2	Medium	Low	70-30%	1211.5	1211.5	5
Small	2	Medium	Medium	30-70%	534	534	6
Small	2	Medium	Medium	50-50%	663	663	3
Small	2	Medium	Medium	70-30%	745.6	745.6	8
Small	3	High	High	30-70%	657.7	657.7	922
Small	3	High	High	50-50%	1059	1059	21
Small	3	High	High	70-30%	1402.5	1402.5	16
Small	3	High	Low	30-70%	1182.3	1182.3	0
Small	3	High	Low	50-50%	1242	1242	0
Small	3	High	Low	70-30%	1334.2	1334.2	0
Small	3	High	Medium	30-70%	1367.7	1367.7	8
Small	3	High	Medium	50-50%	1625.5	1625.5	4
Small	3	High	Medium	70-30%	2017.6	2017.6	3
Small	3	Low	High	30-70%	884.3	884.3	545
Small	3	Low	High	50-50%	1200.5	1200.5	337
Small	3	Low	High	70-30%	1503.3	1503.3	101
Small	3	Low	Low	30-70%	1332.2	1332.2	16
Small	3	Low	Low	50-50%	1294	1294	63
Small	3	Low	Low	70-30%	1830.3	1830.3	49
Small	3	Low	Medium	30-70%	753	753	268
Small	3	Low	Medium	50-50%	1427	1427	118
Small	3	Low	Medium	70-30%	1574.1	1574.1	380
Small	3	Medium	High	30-70%	483	483	5587
Small	3	Medium	High	50-50%	795.5	795.5	11097
Small	3	Medium	High	70-30%	1193.4	1193.4	1532
Small	3	Medium	Low	30-70%	844.1	844.1	2
Small	3	Medium	Low	50-50%	934	934	1
Small	3	Medium	Low	70-30%	1023.6	1023.6	2
Small	3	Medium	Medium	30-70%	1912.2	1912.2	2
Small	3	Medium	Medium	50-50%	2697	2697	3
Small	3	Medium	Medium	70-30%	3231.3	3231.3	3

Table B.2 LBBD results in small examples

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	LBBD		
					LB	UB	Time
Small	1	High	High	30-70%	674	674	44
Small	1	High	High	50-50%	1011	1011	0
Small	1	High	High	70-30%	1285.3	1285.3	2
Small	1	High	Low	30-70%	509	509	0
Small	1	High	Low	50-50%	422.5	422.5	0
Small	1	High	Low	70-30%	548.6	548.6	0
Small	1	High	Medium	30-70%	607.7	607.7	0
Small	1	High	Medium	50-50%	593.5	593.5	1
Small	1	High	Medium	70-30%	634.8	634.8	0
Small	1	Low	High	30-70%	611.3	611.3	56
Small	1	Low	High	50-50%	459	459	75
Small	1	Low	High	70-30%	713.5	713.5	57
Small	1	Low	Low	30-70%	938.9	938.9	11
Small	1	Low	Low	50-50%	1050.5	1050.5	5
Small	1	Low	Low	70-30%	1071.7	1071.7	5
Small	1	Low	Medium	30-70%	322.1	322.1	132
Small	1	Low	Medium	50-50%	479.5	479.5	81
Small	1	Low	Medium	70-30%	605.5	605.5	84
Small	1	Medium	High	30-70%	283.2	283.2	81
Small	1	Medium	High	50-50%	500.5	500.5	40
Small	1	Medium	High	70-30%	681	681	51
Small	1	Medium	Low	30-70%	418.2	418.2	6
Small	1	Medium	Low	50-50%	609	609	6
Small	1	Medium	Low	70-30%	657.6	657.6	7
Small	1	Medium	Medium	30-70%	306.7	306.7	129
Small	1	Medium	Medium	50-50%	409	409	69
Small	1	Medium	Medium	70-30%	575.6	575.6	100
Small	2	High	High	30-70%	855.7	855.7	97
Small	2	High	High	50-50%	1000	1000	81
Small	2	High	High	70-30%	1096.6	1096.6	83
Small	2	High	Low	30-70%	1266.5	1266.5	1
Small	2	High	Low	50-50%	1406.5	1406.5	1
Small	2	High	Low	70-30%	1479.4	1479.4	1
Small	2	High	Medium	30-70%	638.4	638.4	16
Small	2	High	Medium	50-50%	979.5	979.5	35
Small	2	High	Medium	70-30%	1421.1	1421.1	26
Small	2	Low	High	30-70%	462.9	462.9	8870
Small	2	Low	High	50-50%	674	674	23524
Small	2	Low	High	70-30%	954.6	954.6	18448
Small	2	Low	Low	30-70%	1698.9	1698.9	436
Small	2	Low	Low	50-50%	1742	1742	438
Small	2	Low	Low	70-30%	2184.5	2184.5	449

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	LBBD		
					LB	UB	Time
Small	2	Low	Medium	30-70%	1332.1	1742.1	28800
Small	2	Low	Medium	50-50%	1551.5	1957.5	28800
Small	2	Low	Medium	70-30%	1844.4	2244	28800
Small	2	Medium	High	30-70%	559.7	570.3	28800
Small	2	Medium	High	50-50%	886.5	915	28800
Small	2	Medium	High	70-30%	1145.9	1193.4	28800
Small	2	Medium	Low	30-70%	890.1	890.1	9
Small	2	Medium	Low	50-50%	913.5	913.5	4
Small	2	Medium	Low	70-30%	1211.5	1211.5	12
Small	2	Medium	Medium	30-70%	534	534	5
Small	2	Medium	Medium	50-50%	663	663	5
Small	2	Medium	Medium	70-30%	745.6	745.6	8
Small	3	High	High	30-70%	657.7	657.7	22
Small	3	High	High	50-50%	1059	1059	0
Small	3	High	High	70-30%	1402.5	1402.5	2
Small	3	High	Low	30-70%	1182.3	1182.3	0
Small	3	High	Low	50-50%	1242	1242	0
Small	3	High	Low	70-30%	1334.2	1334.2	0
Small	3	High	Medium	30-70%	1367.7	1367.7	0
Small	3	High	Medium	50-50%	1625.5	1625.5	0
Small	3	High	Medium	70-30%	2017.6	2017.6	0
Small	3	Low	High	30-70%	721.8	884.3	28800
Small	3	Low	High	50-50%	1005.5	1200.5	28800
Small	3	Low	High	70-30%	1309.6	1503.3	28800
Small	3	Low	Low	30-70%	1332.2	1332.2	19019
Small	3	Low	Low	50-50%	1294	1294	6104
Small	3	Low	Low	70-30%	1830.3	1830.3	21562
Small	3	Low	Medium	30-70%	746.7	759.9	28800
Small	3	Low	Medium	50-50%	1403	1435	28800
Small	3	Low	Medium	70-30%	1568.9	1582.1	28800
Small	3	Medium	High	30-70%	483	483	12263
Small	3	Medium	High	50-50%	795.5	795.5	8661
Small	3	Medium	High	70-30%	1145.9	1193.4	28800
Small	3	Medium	Low	30-70%	844.1	844.1	1
Small	3	Medium	Low	50-50%	934	934	1
Small	3	Medium	Low	70-30%	1023.6	1023.6	2
Small	3	Medium	Medium	30-70%	1912.2	1912.2	0
Small	3	Medium	Medium	50-50%	2697	2697	0
Small	3	Medium	Medium	70-30%	3231.3	3231.3	0

Table B.3 F-LPR and P-LPR results in small examples

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	F-LPR		P-LPR	
					LB	Time	LB	Time
Small	1	High	High	30-70%	674	3	674	2
Small	1	High	High	50-50%	1011	1	1011	1
Small	1	High	High	70-30%	1285.3	1	1285.3	1
Small	1	High	Low	30-70%	509	0	509	0
Small	1	High	Low	50-50%	422.5	0	422.5	0
Small	1	High	Low	70-30%	548.6	0	548.6	0
Small	1	High	Medium	30-70%	607.7	0	607.7	0
Small	1	High	Medium	50-50%	593.5	0	593.5	0
Small	1	High	Medium	70-30%	634.8	0	634.8	0
Small	1	Low	High	30-70%	607.3	4	607.3	2
Small	1	Low	High	50-50%	459	5	459	6
Small	1	Low	High	70-30%	709.5	3	709.5	3
Small	1	Low	Low	30-70%	938.9	0	938.9	0
Small	1	Low	Low	50-50%	1050.5	0	1050.5	0
Small	1	Low	Low	70-30%	1071.7	0	1071.7	0
Small	1	Low	Medium	30-70%	322.1	1	322.1	5
Small	1	Low	Medium	50-50%	479.5	1	479.5	1
Small	1	Low	Medium	70-30%	605.5	1	605.5	1
Small	1	Medium	High	30-70%	272.8	2	272.8	6
Small	1	Medium	High	50-50%	488	0	488	1
Small	1	Medium	High	70-30%	650.5	3	650.5	0
Small	1	Medium	Low	30-70%	398.2	1	398.2	1
Small	1	Medium	Low	50-50%	589.1	1	589.1	0
Small	1	Medium	Low	70-30%	637.7	0	637.7	0
Small	1	Medium	Medium	30-70%	249.9	0	249.9	0
Small	1	Medium	Medium	50-50%	377.5	1	377.5	0
Small	1	Medium	Medium	70-30%	516.5	2	516.5	1
Small	2	High	High	30-70%	831.7	9	831.7	7
Small	2	High	High	50-50%	976	8	976	4
Small	2	High	High	70-30%	1072.6	4	1072.6	6
Small	2	High	Low	30-70%	1266.5	0	1266.5	0
Small	2	High	Low	50-50%	1406.5	0	1406.5	0
Small	2	High	Low	70-30%	1479.4	0	1479.4	0
Small	2	High	Medium	30-70%	637.2	2	637.2	3
Small	2	High	Medium	50-50%	976.5	4	976.5	5
Small	2	High	Medium	70-30%	1416.9	3	1416.9	2
Small	2	Low	High	30-70%	462.9	30	462.9	42
Small	2	Low	High	50-50%	674	5	674	7
Small	2	Low	High	70-30%	954.6	31	954.6	29
Small	2	Low	Low	30-70%	1698.9	48	1698.9	50
Small	2	Low	Low	50-50%	1742	36	1742	39
Small	2	Low	Low	70-30%	2184.5	34	2184.5	48

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	F-LPR		P-LPR	
					LB	Time	LB	Time
Small	2	Low	Medium	30-70%	1610.1	366	1610.1	448
Small	2	Low	Medium	50-50%	1820.5	320	1820.5	361
Small	2	Low	Medium	70-30%	2130.8	408	2130.8	523
Small	2	Medium	High	30-70%	510.2	28	510.1	26
Small	2	Medium	High	50-50%	836	6	836	7
Small	2	Medium	High	70-30%	1144	204	1108.3	126
Small	2	Medium	Low	30-70%	890.1	7	890.1	10
Small	2	Medium	Low	50-50%	913.5	15	913.5	14
Small	2	Medium	Low	70-30%	1211.5	23	1211.5	14
Small	2	Medium	Medium	30-70%	534	11	534	9
Small	2	Medium	Medium	50-50%	663	10	663	4
Small	2	Medium	Medium	70-30%	745.6	7	745.6	10
Small	3	High	High	30-70%	657.7	11	657.7	17
Small	3	High	High	50-50%	1059	17	1059	15
Small	3	High	High	70-30%	1402.5	14	1402.5	18
Small	3	High	Low	30-70%	1182.3	0	1182.3	0
Small	3	High	Low	50-50%	1242	0	1242	0
Small	3	High	Low	70-30%	1334.2	0	1334.2	0
Small	3	High	Medium	30-70%	1367.7	1	1367.7	1
Small	3	High	Medium	50-50%	1625.5	2	1625.5	2
Small	3	High	Medium	70-30%	2017.6	2	2017.6	1
Small	3	Low	High	30-70%	884.3	423	884.3	664
Small	3	Low	High	50-50%	1200.5	1155	1200.5	1607
Small	3	Low	High	70-30%	1503.3	1055	1503.3	624
Small	3	Low	Low	30-70%	1332.2	59	1332.2	45
Small	3	Low	Low	50-50%	1294	52	1294	188
Small	3	Low	Low	70-30%	1830.3	170	1830.3	129
Small	3	Low	Medium	30-70%	753	623	753	335
Small	3	Low	Medium	50-50%	1427	265	1399	14
Small	3	Low	Medium	70-30%	1574.1	759	1574.1	1232
Small	3	Medium	High	30-70%	466.8	48	462.9	9
Small	3	Medium	High	50-50%	749	15	749	17
Small	3	Medium	High	70-30%	1091	86	1091	61
Small	3	Medium	Low	30-70%	844.1	1	844.1	2
Small	3	Medium	Low	50-50%	934	2	934	2
Small	3	Medium	Low	70-30%	1023.6	1	1023.6	2
Small	3	Medium	Medium	30-70%	1912.2	1	1912.2	2
Small	3	Medium	Medium	50-50%	2697	1	2697	3
Small	3	Medium	Medium	70-30%	3231.3	1	3231.3	2

Table B.4 search algorithms results in small examples

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	TS-TCL		TS-CL		TS-IL	
					UB	Time	UB	Time	UB	Time
Small	1	High	High	30-70%	683	33	691.1	1	691.1	1
Small	1	High	High	50-50%	1011	16	1011	1	1011	0
Small	1	High	High	70-30%	1285.3	0	1299.3	15	1285.3	1
Small	1	High	Low	30-70%	509	2	509	1	509	0
Small	1	High	Low	50-50%	422.5	1	422.5	3	422.5	0
Small	1	High	Low	70-30%	548.6	2	548.6	1	548.6	0
Small	1	High	Medium	30-70%	607.7	24	613.7	15	607.7	15
Small	1	High	Medium	50-50%	599.5	21	593.5	31	593.5	18
Small	1	High	Medium	70-30%	640.8	18	634.8	31	634.8	12
Small	1	Low	High	30-70%	652.3	13	611.3	2	703.3	3
Small	1	Low	High	50-50%	459	47	459	7	459	3
Small	1	Low	High	70-30%	749.5	20	713.5	4	713.5	3
Small	1	Low	Low	30-70%	942.9	5	938.9	2	938.9	1
Small	1	Low	Low	50-50%	1054.5	10	1050.5	2	1050.5	1
Small	1	Low	Low	70-30%	1075.7	5	1071.7	2	1071.7	1
Small	1	Low	Medium	30-70%	341.5	52	322.1	0	322.1	0
Small	1	Low	Medium	50-50%	512.5	74	507.5	0	507.5	0
Small	1	Low	Medium	70-30%	643.7	78	633.5	0	633.5	0
Small	1	Medium	High	30-70%	285	16	283.3	0	283.3	0
Small	1	Medium	High	50-50%	503.5	29	505.5	0	505.5	0
Small	1	Medium	High	70-30%	681	12	681	0	681	0
Small	1	Medium	Low	30-70%	418.2	11	418.2	1	419.2	0
Small	1	Medium	Low	50-50%	609	12	609	1	610	0
Small	1	Medium	Low	70-30%	657.6	10	657.6	1	658.6	0
Small	1	Medium	Medium	30-70%	309.3	21	312.9	7	309.3	4
Small	1	Medium	Medium	50-50%	409	39	409	10	413.5	3
Small	1	Medium	Medium	70-30%	579.5	42	579.5	11	579.5	7
Small	2	High	High	30-70%	855.7	24	855.7	4	869.7	2
Small	2	High	High	50-50%	1000	21	1000	5	1014	2
Small	2	High	High	70-30%	1096.6	29	1096.6	4	1110.6	2
Small	2	High	Low	30-70%	1266.5	10	1266.5	0	1266.5	0
Small	2	High	Low	50-50%	1406.5	9	1406.5	0	1406.5	1
Small	2	High	Low	70-30%	1479.4	12	1479.4	0	1479.4	0
Small	2	High	Medium	30-70%	638.4	33	638.4	10	638.4	7
Small	2	High	Medium	50-50%	997.5	37	997.5	13	997.5	7
Small	2	High	Medium	70-30%	1467	44	1467	11	1467	8
Small	2	Low	High	30-70%	473.7	487	468.3	35	473.7	22
Small	2	Low	High	50-50%	674	470	695.5	185	854.4	8
Small	2	Low	High	70-30%	954.6	495	1033.2	109	1049.8	31
Small	2	Low	Low	30-70%	1729.9	379	1698.9	71	1716.9	17

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	TS-TCL		TS-CL		TS-IL	
					UB	Time	UB	Time	UB	Time
Small	2	Low	Low	50-50%	1742	389	1742	86	1760	17
Small	2	Low	Low	70-30%	2184.5	469	2184.5	81	2202.5	19
Small	2	Low	Medium	30-70%	1618.1	541	1642.1	29	1726.1	7
Small	2	Low	Medium	50-50%	1820.5	792	1835.5	33	1872.2	7
Small	2	Low	Medium	70-30%	2227	861	2130.8	19	2312	4
Small	2	Medium	High	30-70%	570.3	630	581.2	5	581.2	3
Small	2	Medium	High	50-50%	915	720	948.5	3	948.5	3
Small	2	Medium	High	70-30%	1193.4	756	1193.9	4	1193.9	3
Small	2	Medium	Low	30-70%	890.1	55	890.1	5	1004.1	0
Small	2	Medium	Low	50-50%	913.5	80	913.5	1	1008.5	0
Small	2	Medium	Low	70-30%	1211.5	79	1211.5	2	1319.8	0
Small	2	Medium	Medium	30-70%	548	17	534	1	534	0
Small	2	Medium	Medium	50-50%	663	14	663	1	663	0
Small	2	Medium	Medium	70-30%	759	15	745.6	4	745.6	0
Small	3	High	High	30-70%	657.7	12	657.7	32	657.7	1
Small	3	High	High	50-50%	1059	13	1142.5	28	1267.5	1
Small	3	High	High	70-30%	1402.5	12	1533.4	28	1647.9	1
Small	3	High	Low	30-70%	1182.3	10	1220.3	1	1220.3	0
Small	3	High	Low	50-50%	1242	15	1242	0	1242	0
Small	3	High	Low	70-30%	1334.2	16	1370.4	0	1370.4	0
Small	3	High	Medium	30-70%	1367.7	133	1367.7	2	1367.7	0
Small	3	High	Medium	50-50%	1625.5	207	1625.5	1	1625.5	3
Small	3	High	Medium	70-30%	2017.6	171	2017.6	1	2017.6	1
Small	3	Low	High	30-70%	969.8	1506	930.8	67	981.8	24
Small	3	Low	High	50-50%	1200.5	1230	1277.5	148	1635.5	32
Small	3	Low	High	70-30%	1637.1	3274	1575.8	144	2006.3	36
Small	3	Low	Low	30-70%	1352.2	507	1372.2	12	1332.2	6
Small	3	Low	Low	50-50%	1308	622	1296	17	1294	10
Small	3	Low	Low	70-30%	1841.3	586	1861.3	16	1830.3	10
Small	3	Low	Medium	30-70%	753	590	797.6	115	1010.9	1
Small	3	Low	Medium	50-50%	1427	780	1435.5	159	1496.5	4
Small	3	Low	Medium	70-30%	1574.1	865	1598.4	184	1980.4	3
Small	3	Medium	High	30-70%	483	345	526	61	526	4
Small	3	Medium	High	50-50%	795.5	478	866	86	866	4
Small	3	Medium	High	70-30%	1193.4	456	1197.9	63	1389.9	5
Small	3	Medium	Low	30-70%	844.1	657	895.1	64	844.1	9
Small	3	Medium	Low	50-50%	934	245	934	14	934	2
Small	3	Medium	Low	70-30%	1074.6	77	1023.6	16	1023.6	4
Small	3	Medium	Medium	30-70%	1912.2	118	1912.2	4	1912.2	1
Small	3	Medium	Medium	50-50%	2697	223	2697	5	2714.5	3
Small	3	Medium	Medium	70-30%	3231.3	263	3231.3	6	3255.8	2

APPENDIX C. Detailed Results from Medium Size Problems

Table C.1 Branch-and-bound results in medium examples

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	B&B		
					LB	UB	Time
Medium	1	High	High	30-70%	555	625.8	28800
Medium	1	High	High	50-50%	938.2	1038	28800
Medium	1	High	High	70-30%	1291	1477.1	28800
Medium	1	High	Low	30-70%	1076	1076	19
Medium	1	High	Low	50-50%	1745.5	1745.5	45
Medium	1	High	Low	70-30%	2263	2263	33
Medium	1	High	Medium	30-70%	2275.8	2275.8	3
Medium	1	High	Medium	50-50%	3229	3229	4
Medium	1	High	Medium	70-30%	3878.2	3878.2	4
Medium	1	Low	High	30-70%	734.3	1170.7	28800
Medium	1	Low	High	50-50%	1248.1	1728	28800
Medium	1	Low	High	70-30%	1764.7	2442.3	28800
Medium	1	Low	Low	30-70%	1844.1	1948.5	28800
Medium	1	Low	Low	50-50%	2072.5	2072.5	3620
Medium	1	Low	Low	70-30%	2800.8	2912.5	28800
Medium	1	Low	Medium	30-70%	1393.2	1393.2	8385
Medium	1	Low	Medium	50-50%	2048.5	2048.5	1290
Medium	1	Low	Medium	70-30%	2429	2429	4673
Medium	1	Medium	High	30-70%	669.3	991.2	28800
Medium	1	Medium	High	50-50%	1141.4	1492	28800
Medium	1	Medium	High	70-30%	1541.4	2054.1	28800
Medium	1	Medium	Low	30-70%	1400.4	1407.6	28800
Medium	1	Medium	Low	50-50%	1777.5	1783.5	28800
Medium	1	Medium	Low	70-30%	2403.3	2409.3	28800
Medium	1	Medium	Medium	30-70%	1474.9	1474.9	51
Medium	1	Medium	Medium	50-50%	2484	2484	24
Medium	1	Medium	Medium	70-30%	2946.8	2946.8	46
Medium	2	High	High	30-70%	1009.9	1317.9	28800
Medium	2	High	High	50-50%	1681.9	2117.5	28800
Medium	2	High	High	70-30%	2397.1	3067.4	28800
Medium	2	High	Low	30-70%	1097.2	1202.4	28800
Medium	2	High	Low	50-50%	1800.5	1991.5	28800
Medium	2	High	Low	70-30%	2518.8	2800.8	28800
Medium	2	High	Medium	30-70%	875.1	983.2	28800
Medium	2	High	Medium	50-50%	1539.5	1570	28800
Medium	2	High	Medium	70-30%	2065.3	2121.4	28800
Medium	2	Low	High	30-70%	727.1	1292.6	28800
Medium	2	Low	High	50-50%	1241.3	2022.5	28800
Medium	2	Low	High	70-30%	1767.5	2822.9	28800
Medium	2	Low	Low	30-70%	727.2	884.8	28800
Medium	2	Low	Low	50-50%	1230.2	1421	28800

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	B&B		
					LB	UB	Time
Medium	2	Low	Low	70-30%	1678.2	2047.9	28800
Medium	2	Low	Medium	30-70%	629.5	1067.4	28800
Medium	2	Low	Medium	50-50%	1089	1835.5	28800
Medium	2	Low	Medium	70-30%	1529.7	2522.2	28800
Medium	2	Medium	High	30-70%	969.9	1145.1	28800
Medium	2	Medium	High	50-50%	1565.2	1996	28800
Medium	2	Medium	High	70-30%	2122.1	2865.1	28800
Medium	2	Medium	Low	30-70%	884.1	964.2	28800
Medium	2	Medium	Low	50-50%	1085	1500.5	28800
Medium	2	Medium	Low	70-30%	1653.9	1790	28800
Medium	2	Medium	Medium	30-70%	909.6	933	28800
Medium	2	Medium	Medium	50-50%	1512	1553.5	28800
Medium	2	Medium	Medium	70-30%	2144.8	2184	28800
Medium	3	High	High	30-70%	1060.6	1420.8	28800
Medium	3	High	High	50-50%	1786.4	2103	28800
Medium	3	High	High	70-30%	2526.2	2957.5	28800
Medium	3	High	Low	30-70%	2013.2	2013.2	926
Medium	3	High	Low	50-50%	2487	2487	113
Medium	3	High	Low	70-30%	3009	3009	136
Medium	3	High	Medium	30-70%	1127.9	1497.9	28800
Medium	3	High	Medium	50-50%	1880.6	2499.5	28800
Medium	3	High	Medium	70-30%	2632.2	3661.3	28800
Medium	3	Low	High	30-70%	889.4	2070.9	28800
Medium	3	Low	High	50-50%	1479.7	2823.5	28800
Medium	3	Low	High	70-30%	2097.2	3578.3	28800
Medium	3	Low	Low	30-70%	1661.9	1935.7	28800
Medium	3	Low	Low	50-50%	1896.5	2064.5	28800
Medium	3	Low	Low	70-30%	2401	2649.2	28800
Medium	3	Low	Medium	30-70%	2749	3645.2	28800
Medium	3	Low	Medium	50-50%	2681.9	3340	28800
Medium	3	Low	Medium	70-30%	3437.8	4448.1	28800
Medium	3	Medium	High	30-70%	1016.3	1619.1	28800
Medium	3	Medium	High	50-50%	1686.8	2645.5	28800
Medium	3	Medium	High	70-30%	2330.2	3797.5	28800
Medium	3	Medium	Low	30-70%	1922	2119.9	28800
Medium	3	Medium	Low	50-50%	2258.1	2455	28800
Medium	3	Medium	Low	70-30%	2826.8	3249.6	28800
Medium	3	Medium	Medium	30-70%	836.8	1360.5	28800
Medium	3	Medium	Medium	50-50%	1399.4	2277	28800
Medium	3	Medium	Medium	70-30%	1965.1	2947.3	28800

Table C.2 LBBD results in medium examples

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	LBBD		
					LB	UB	Time
Medium	1	High	High	30-70%	579.9	660.3	28800
Medium	1	High	High	50-50%	966.5	1096.5	28800
Medium	1	High	High	70-30%	1353.3	1551.7	28800
Medium	1	High	Low	30-70%	1076	1076	5
Medium	1	High	Low	50-50%	1745.5	1745.5	15930
Medium	1	High	Low	70-30%	2256.7	2263	28800
Medium	1	High	Medium	30-70%	2275.8	2275.8	0
Medium	1	High	Medium	50-50%	3229	3229	1
Medium	1	High	Medium	70-30%	3878.2	3878.2	0
Medium	1	Low	High	30-70%	719.7	2012.3	28800
Medium	1	Low	High	50-50%	1199.5	2801	28800
Medium	1	Low	High	70-30%	1679.3	4080.2	28800
Medium	1	Low	Low	30-70%	1547.9	2449.9	28800
Medium	1	Low	Low	50-50%	1835.5	2426	28800
Medium	1	Low	Low	70-30%	2644.2	3156.3	28800
Medium	1	Low	Medium	30-70%	1334	1777.4	28800
Medium	1	Low	Medium	50-50%	1962	2202.5	28800
Medium	1	Low	Medium	70-30%	2379.6	2493.4	28800
Medium	1	Medium	High	30-70%	751.5	1158.4	28800
Medium	1	Medium	High	50-50%	1236.5	1475.5	28800
Medium	1	Medium	High	70-30%	1729	2013.4	28800
Medium	1	Medium	Low	30-70%	1400.4	1474	28800
Medium	1	Medium	Low	50-50%	1776.5	1850	28800
Medium	1	Medium	Low	70-30%	2316.3	2409.3	28800
Medium	1	Medium	Medium	30-70%	1474.9	1474.9	25
Medium	1	Medium	Medium	50-50%	2484	2484	41
Medium	1	Medium	Medium	70-30%	2946.8	2946.8	4
Medium	2	High	High	30-70%	1139.1	1329.3	28800
Medium	2	High	High	50-50%	1898.5	2215.5	28800
Medium	2	High	High	70-30%	2657.9	3101.7	28800
Medium	2	High	Low	30-70%	1059	1361.1	28800
Medium	2	High	Low	50-50%	1765	2271	28800
Medium	2	High	Low	70-30%	2471	3167.5	28800
Medium	2	High	Medium	30-70%	875.1	1054.8	28800
Medium	2	High	Medium	50-50%	1539.5	1655.5	28800
Medium	2	High	Medium	70-30%	2065.3	2254.9	28800
Medium	2	Low	High	30-70%	849	1563.9	28800
Medium	2	Low	High	50-50%	1415	2856.5	28800
Medium	2	Low	High	70-30%	1981	3766.7	28800
Medium	2	Low	Low	30-70%	670.5	932.6	28800
Medium	2	Low	Low	50-50%	1119	1454	28800
Medium	2	Low	Low	70-30%	1573.9	2028.1	28800

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	LBBD		
					LB	UB	Time
Medium	2	Low	Medium	30-70%	789.6	1054	28800
Medium	2	Low	Medium	50-50%	1316	1804.5	28800
Medium	2	Low	Medium	70-30%	1842.4	2473.8	28800
Medium	2	Medium	High	30-70%	1035.6	1263.9	28800
Medium	2	Medium	High	50-50%	1726	1968.5	28800
Medium	2	Medium	High	70-30%	2416.4	2811.2	28800
Medium	2	Medium	Low	30-70%	854.3	1001.8	28800
Medium	2	Medium	Low	50-50%	1305.5	1426	28800
Medium	2	Medium	Low	70-30%	1625.2	1766.1	28800
Medium	2	Medium	Medium	30-70%	902.7	939.6	28800
Medium	2	Medium	Medium	50-50%	1504.5	1566	28800
Medium	2	Medium	Medium	70-30%	2085.3	2277.1	28800
Medium	3	High	High	30-70%	1247.5	1379.8	28800
Medium	3	High	High	50-50%	2024.5	2131.5	28800
Medium	3	High	High	70-30%	2830.1	3025.6	28800
Medium	3	High	Low	30-70%	2013.2	2013.2	110
Medium	3	High	Low	50-50%	2487	2487	36
Medium	3	High	Low	70-30%	3009	3009	27
Medium	3	High	Medium	30-70%	1422.9	1601.9	28800
Medium	3	High	Medium	50-50%	2390.5	2705	28800
Medium	3	High	Medium	70-30%	3389.8	3896.6	28800
Medium	3	Low	High	30-70%	1019.4	2545.3	28800
Medium	3	Low	High	50-50%	1699	3736	28800
Medium	3	Low	High	70-30%	2380.1	4291.9	28800
Medium	3	Low	Low	30-70%	1553.4	2025.1	28800
Medium	3	Low	Low	50-50%	1751	2181.5	28800
Medium	3	Low	Low	70-30%	2243.1	2687.1	28800
Medium	3	Low	Medium	30-70%	2546	4583.1	28800
Medium	3	Low	Medium	50-50%	2578.5	4221	28800
Medium	3	Low	Medium	70-30%	3287.4	5155.8	28800
Medium	3	Medium	High	30-70%	1226.1	1582.8	28800
Medium	3	Medium	High	50-50%	2043.5	2674	28800
Medium	3	Medium	High	70-30%	2860.9	3759	28800
Medium	3	Medium	Low	30-70%	1897.9	2150.8	28800
Medium	3	Medium	Low	50-50%	2214.5	2707.5	28800
Medium	3	Medium	Low	70-30%	2770.8	3302.3	28800
Medium	3	Medium	Medium	30-70%	1095.6	1263	28800
Medium	3	Medium	Medium	50-50%	1858.5	2398.5	28800
Medium	3	Medium	Medium	70-30%	2546.1	2841	28800

Table C.3 F-LPR and P-LPR results in medium examples

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	F-LPR		P-LPR	
					LB	Time	LB	Time
Medium	1	High	High	30-70%	553	28800	582	28800
Medium	1	High	High	50-50%	921.6	28800	972.5	28800
Medium	1	High	High	70-30%	1290.2	28800	1357.1	28800
Medium	1	High	Low	30-70%	1076	22	1076	22
Medium	1	High	Low	50-50%	1745.5	34	1745.5	42
Medium	1	High	Low	70-30%	2263	24	2263	29
Medium	1	High	Medium	30-70%	2266.8	4	2266.8	4
Medium	1	High	Medium	50-50%	3214	4	3214	2
Medium	1	High	Medium	70-30%	3857.2	4	3857.2	3
Medium	1	Low	High	30-70%	987.5	28800	982.1	28800
Medium	1	Low	High	50-50%	1554.4	28800	1565.5	28800
Medium	1	Low	High	70-30%	2190.6	28800	2240.6	28800
Medium	1	Low	Low	30-70%	1948.5	28800	1941.5	10112
Medium	1	Low	Low	50-50%	2072.5	5412	2072.5	1860
Medium	1	Low	Low	70-30%	2897	9580	2895.5	9634
Medium	1	Low	Medium	30-70%	1393.2	13923	1381.8	28800
Medium	1	Low	Medium	50-50%	2048.5	12538	2048.5	5183
Medium	1	Low	Medium	70-30%	2429	5606	2399.6	1860
Medium	1	Medium	High	30-70%	774.3	28800	760.8	2323
Medium	1	Medium	High	50-50%	1233.3	28800	1262.5	1461
Medium	1	Medium	High	70-30%	1748.1	28800	1765.4	2110
Medium	1	Medium	Low	30-70%	1400.4	175	1400.4	188
Medium	1	Medium	Low	50-50%	1776.5	1806	1776.5	66
Medium	1	Medium	Low	70-30%	2385	28800	2316.3	88
Medium	1	Medium	Medium	30-70%	1474.9	54	1474.9	30
Medium	1	Medium	Medium	50-50%	2484	244	2484	13
Medium	1	Medium	Medium	70-30%	2946.8	26	2946.8	21
Medium	2	High	High	30-70%	1083.3	28800	1142.4	4056
Medium	2	High	High	50-50%	1800.2	28800	1904.1	2025
Medium	2	High	High	70-30%	2527.7	28800	2665.6	8084
Medium	2	High	Low	30-70%	1083.3	1378	1073.4	294
Medium	2	High	Low	50-50%	1805.4	2830	1789	145
Medium	2	High	Low	70-30%	2510.2	3746	2504.6	206
Medium	2	High	Medium	30-70%	743.4	28800	875.1	202
Medium	2	High	Medium	50-50%	1259.5	28800	1539.5	1473
Medium	2	High	Medium	70-30%	1730.5	28800	2065.3	486
Medium	2	Low	High	30-70%	882	28800	921.6	28800
Medium	2	Low	High	50-50%	1459.5	28800	1472.3	28800
Medium	2	Low	High	70-30%	2125.9	28800	1997.2	28800
Medium	2	Low	Low	30-70%	801.6	28800	833.4	28800
Medium	2	Low	Low	50-50%	1402.5	28800	1357.5	28800
Medium	2	Low	Low	70-30%	2022.5	28800	1929.5	28800

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	F-LPR		P-LPR	
					LB	Time	LB	Time
Medium	2	Low	Medium	30-70%	684.4	28800	1025.7	28800
Medium	2	Low	Medium	50-50%	1409.5	28800	1396.5	28800
Medium	2	Low	Medium	70-30%	2117.5	28800	1887.8	28800
Medium	2	Medium	High	30-70%	986.7	28800	1054.2	287
Medium	2	Medium	High	50-50%	1696.6	28800	1687	28800
Medium	2	Medium	High	70-30%	2432.5	28800	2365.3	28800
Medium	2	Medium	Low	30-70%	598.8	28800	891.1	28800
Medium	2	Medium	Low	50-50%	965.2	28800	1365	28800
Medium	2	Medium	Low	70-30%	1198.9	28800	1709.8	28800
Medium	2	Medium	Medium	30-70%	911.7	956	911.7	162
Medium	2	Medium	Medium	50-50%	1519.5	4429	1519.5	989
Medium	2	Medium	Medium	70-30%	2107.7	889	2107.7	653
Medium	3	High	High	30-70%	1143.8	28800	1247.5	7909
Medium	3	High	High	50-50%	2024.5	7876	2024.5	3316
Medium	3	High	High	70-30%	2835.7	3870	2835.7	8210
Medium	3	High	Low	30-70%	2013.2	482	2005	123
Medium	3	High	Low	50-50%	2487	166	2487	185
Medium	3	High	Low	70-30%	3009	212	3009	370
Medium	3	High	Medium	30-70%	1422.9	180	1422.9	182
Medium	3	High	Medium	50-50%	2390.5	177	2390.5	86
Medium	3	High	Medium	70-30%	3399.7	28800	3389.8	197
Medium	3	Low	High	30-70%	870.9	28	1184	28800
Medium	3	Low	High	50-50%	2140.2	28800	1962.8	28800
Medium	3	Low	High	70-30%	2830.6	28800	2925.9	28800
Medium	3	Low	Low	30-70%	1715.9	28800	1644.6	28800
Medium	3	Low	Low	50-50%	1874.6	28800	2038	28800
Medium	3	Low	Low	70-30%	2389.4	28800	2535.3	28800
Medium	3	Low	Medium	30-70%	3160.2	28800	3212.2	28800
Medium	3	Low	Medium	50-50%	2987.1	28800	2987.1	28800
Medium	3	Low	Medium	70-30%	3875.5	28800	3886.4	28800
Medium	3	Medium	High	30-70%	1109.2	28800	1281.9	28800
Medium	3	Medium	High	50-50%	2001.1	28800	1968.5	28800
Medium	3	Medium	High	70-30%	2872.3	28800	2839.9	28800
Medium	3	Medium	Low	30-70%	1948.7	2469	1948.7	4279
Medium	3	Medium	Low	50-50%	2300	28800	2429.7	28800
Medium	3	Medium	Low	70-30%	2872.2	9377	2848.3	28800
Medium	3	Medium	Medium	30-70%	1170.7	28800	1095.6	462
Medium	3	Medium	Medium	50-50%	1835.1	28800	1895.5	28800
Medium	3	Medium	Medium	70-30%	2711.3	28800	2546.1	28800

Table C.4 Search algorithms results in medium examples

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	TS-CL		TS-IL	
					UB	Time	UB	Time
Medium	1	High	High	30-70%	625.5	3349	625.5	1155
Medium	1	High	High	50-50%	1055	4541	1043	1371
Medium	1	High	High	70-30%	1535.7	4326	1561.6	858
Medium	1	High	Low	30-70%	1133.9	7	1143.3	3
Medium	1	High	Low	50-50%	1887.5	10	1887.5	5
Medium	1	High	Low	70-30%	2269.1	11	2274	5
Medium	1	High	Medium	30-70%	2275.8	1	2275.8	0
Medium	1	High	Medium	50-50%	3229	1	3229	0
Medium	1	High	Medium	70-30%	3878.2	0	3878.2	0
Medium	1	Low	High	30-70%	1298.5	1271	1388.9	15
Medium	1	Low	High	50-50%	1822.5	1598	1886.5	56
Medium	1	Low	High	70-30%	2661.8	1485	3396.8	34
Medium	1	Low	Low	30-70%	1968.5	46	2068.5	53
Medium	1	Low	Low	50-50%	2079.5	87	2149.5	48
Medium	1	Low	Low	70-30%	3029.8	48	3029.8	38
Medium	1	Low	Medium	30-70%	1406.5	2222	1500.3	40
Medium	1	Low	Medium	50-50%	2053.5	2678	2187	79
Medium	1	Low	Medium	70-30%	2551	2425	3111	56
Medium	1	Medium	High	30-70%	999.5	661	1088.3	28
Medium	1	Medium	High	50-50%	1531.5	923	1500	222
Medium	1	Medium	High	70-30%	2042.2	839	2062.3	189
Medium	1	Medium	Low	30-70%	1485.8	1096	1532.6	34
Medium	1	Medium	Low	50-50%	1816.5	1082	2025	92
Medium	1	Medium	Low	70-30%	2442.5	933	2488.5	79
Medium	1	Medium	Medium	30-70%	1474.9	271	1487.5	26
Medium	1	Medium	Medium	50-50%	2484	518	2692.5	67
Medium	1	Medium	Medium	70-30%	2946.8	434	2976.2	50
Medium	2	High	High	30-70%	1447.8	173	1410.6	1069
Medium	2	High	High	50-50%	2264	1258	2188.5	1779
Medium	2	High	High	70-30%	3309.6	1570	3263.4	1685
Medium	2	High	Low	30-70%	1244.3	2437	1243.4	571
Medium	2	High	Low	50-50%	2067.5	4081	2076.5	1110
Medium	2	High	Low	70-30%	2801.7	3037	2847.2	868
Medium	2	High	Medium	30-70%	990.3	586	1001.2	46
Medium	2	High	Medium	50-50%	1611	2256	1750.5	114
Medium	2	High	Medium	70-30%	2280.9	330	2280.9	73
Medium	2	Low	High	30-70%	1334.3	4395	1447.5	80
Medium	2	Low	High	50-50%	2149.5	6051	2026	1593
Medium	2	Low	High	70-30%	2944.6	4513	3068.3	172
Medium	2	Low	Low	30-70%	983.2	3372	966.5	3307
Medium	2	Low	Low	50-50%	1488.5	5763	1462.5	3829
Medium	2	Low	Low	70-30%	2146.7	3850	2292.2	471

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	TS-CL		TS-IL	
					UB	Time	UB	Time
Medium	2	Low	Medium	30-70%	1203.1	4734	1156.4	4152
Medium	2	Low	Medium	50-50%	1653	6541	1733.5	4703
Medium	2	Low	Medium	70-30%	2468	5022	2332.9	3840
Medium	2	Medium	High	30-70%	1195.8	4241	1482.9	84
Medium	2	Medium	High	50-50%	1991.5	6216	2048	163
Medium	2	Medium	High	70-30%	2925.3	4460	3528.7	128
Medium	2	Medium	Low	30-70%	1045.6	2816	1112.6	591
Medium	2	Medium	Low	50-50%	1426	3697	1471	2018
Medium	2	Medium	Low	70-30%	1904.1	1996	2290.1	28
Medium	2	Medium	Medium	30-70%	933	1831	1064.5	139
Medium	2	Medium	Medium	50-50%	1669	3448	1775	415
Medium	2	Medium	Medium	70-30%	2217.2	2646	2210	87
Medium	3	High	High	30-70%	1357	1140	1638.2	22
Medium	3	High	High	50-50%	2116.4	1439	2169.5	41
Medium	3	High	High	70-30%	3031	697	3047.8	48
Medium	3	High	Low	30-70%	2097.2	244	2146.4	53
Medium	3	High	Low	50-50%	2487	835	2524	363
Medium	3	High	Low	70-30%	3101.7	708	3123.1	150
Medium	3	High	Medium	30-70%	1490.7	2372	1612.6	58
Medium	3	High	Medium	50-50%	2546.5	12008	2621.5	313
Medium	3	High	Medium	70-30%	3722	2619	4026	136
Medium	3	Low	High	30-70%	2078	1317	2549.9	44
Medium	3	Low	High	50-50%	2876.5	1564	3829	479
Medium	3	Low	High	70-30%	3620.4	367	4080.5	172
Medium	3	Low	Low	30-70%	1986	4339	2025.4	62
Medium	3	Low	Low	50-50%	2219.5	4733	2558.5	165
Medium	3	Low	Low	70-30%	2878.7	3662	3495.7	155
Medium	3	Low	Medium	30-70%	3686.4	385	4077	43
Medium	3	Low	Medium	50-50%	3310.5	2462	3331.5	179
Medium	3	Low	Medium	70-30%	4523.4	394	4977.4	84
Medium	3	Medium	High	30-70%	1705.5	3418	2055.6	90
Medium	3	Medium	High	50-50%	2627	4720	2765.5	485
Medium	3	Medium	High	70-30%	3806.4	4317	3956.8	152
Medium	3	Medium	Low	30-70%	2177	435	2199	117
Medium	3	Medium	Low	50-50%	2475	824	2849	555
Medium	3	Medium	Low	70-30%	3189.1	465	3267.5	250
Medium	3	Medium	Medium	30-70%	1321.1	2107	1782.1	77
Medium	3	Medium	Medium	50-50%	2207	3985	2462.5	502
Medium	3	Medium	Medium	70-30%	3074.3	2133	3074	312

APPENDIX D. Detailed Results from Large Size Problems

Table D.1 Branch-and-bound results in large examples

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	B&B		
					LB	UB	Time
Large	1	High	High	30-70%	1129.8	1650.3	28800
Large	1	High	High	50-50%	1893.5	3000	28800
Large	1	High	High	70-30%	2641.5	4041.1	28800
Large	1	High	Low	30-70%	2413.8	4483.1	28800
Large	1	High	Low	50-50%	2685.5	3880.5	28800
Large	1	High	Low	70-30%	3760.6	5186.4	28800
Large	1	High	Medium	30-70%	1238.2	1877.1	28800
Large	1	High	Medium	50-50%	2041.8	2900	28800
Large	1	High	Medium	70-30%	2895.3	3951.5	28800
Large	1	Low	High	30-70%	906.4	1932.9	28800
Large	1	Low	High	50-50%	1583.3	2853	28800
Large	1	Low	High	70-30%	2140.4	4122.3	28800
Large	1	Low	Low	30-70%	1860.9	4268.2	28800
Large	1	Low	Low	50-50%	3203.3	4991.5	28800
Large	1	Low	Low	70-30%	3963.3	7171.6	28800
Large	1	Low	Medium	30-70%	647.9	1947.9	28800
Large	1	Low	Medium	50-50%	1072.3	3087.5	28800
Large	1	Low	Medium	70-30%	1521.7	3429.7	28800
Large	1	Medium	High	30-70%	1212.9	2217	28800
Large	1	Medium	High	50-50%	2009.4	3856.1	28800
Large	1	Medium	High	70-30%	2802.4	4959.5	28800
Large	1	Medium	Low	30-70%	2931.1	4147.3	28800
Large	1	Medium	Low	50-50%	4073	5764.5	28800
Large	1	Medium	Low	70-30%	5098.5	6123.4	28800
Large	1	Medium	Medium	30-70%	1485.3	2508.2	28800
Large	1	Medium	Medium	50-50%	2512.8	3777.5	28800
Large	1	Medium	Medium	70-30%	3485.6	5500.5	28800
Large	2	High	High	30-70%	1187.4	2314.8	28800
Large	2	High	High	50-50%	1994.6	3477.5	28800
Large	2	High	High	70-30%	2738.4	4441.5	28800
Large	2	High	Low	30-70%	1289.4	2181.1	28800
Large	2	High	Low	50-50%	2317.2	3214	28800
Large	2	High	Low	70-30%	3459.8	4306.8	28800
Large	2	High	Medium	30-70%	1179.2	2101.8	28800
Large	2	High	Medium	50-50%	2027.5	3324.5	28800
Large	2	High	Medium	70-30%	2764.9	4786.9	28800
Large	2	Low	High	30-70%	1322.8	2491.5	28800
Large	2	Low	High	50-50%	2187	3730	28800
Large	2	Low	High	70-30%	3124	5971.3	28800
Large	2	Low	Low	30-70%	1330.4	3381.9	28800
Large	2	Low	Low	50-50%	2112.6	4871.5	28800

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	B&B		
					LB	UB	Time
Large	2	Low	Low	70-30%	2639.4	5728.8	28800
Large	2	Low	Medium	30-70%	1179.1	2134.8	28800
Large	2	Low	Medium	50-50%	2606.7	5666	28800
Large	2	Low	Medium	70-30%	3603.5	7435.4	28800
Large	2	Medium	High	30-70%	1515.2	3026.7	28800
Large	2	Medium	High	50-50%	2508.1	5273.5	28800
Large	2	Medium	High	70-30%	3521.2	7694.4	28800
Large	2	Medium	Low	30-70%	2891.5	4985.3	28800
Large	2	Medium	Low	50-50%	3559.8	5615	28800
Large	2	Medium	Low	70-30%	3893.3	6003	28800
Large	2	Medium	Medium	30-70%	975.2	1603.9	28800
Large	2	Medium	Medium	50-50%	1609.2	2478	28800
Large	2	Medium	Medium	70-30%	2291.6	3454.5	28800
Large	3	High	High	30-70%	1649.2	4232.4	28800
Large	3	High	High	50-50%	2770.9	7459	28800
Large	3	High	High	70-30%	3915.9	9379.3	28800
Large	3	High	Low	30-70%	1745.2	2470.7	28800
Large	3	High	Low	50-50%	2837.5	3657.5	28800
Large	3	High	Low	70-30%	3702.2	4789.5	28800
Large	3	High	Medium	30-70%	1370.6	2855.4	28800
Large	3	High	Medium	50-50%	2280.1	4581.5	28800
Large	3	High	Medium	70-30%	3209.6	6586.3	28800
Large	3	Low	High	30-70%	1821.3	5230.8	28800
Large	3	Low	High	50-50%	3004.2	6822	28800
Large	3	Low	High	70-30%	4265.7	10938.5	28800
Large	3	Low	Low	30-70%	2409	3181.5	28800
Large	3	Low	Low	50-50%	3897.7	5153	28800
Large	3	Low	Low	70-30%	4946.7	6751	28800
Large	3	Low	Medium	30-70%	1360	3831.6	28800
Large	3	Low	Medium	50-50%	2211.2	6046	28800
Large	3	Low	Medium	70-30%	3135.4	7567.4	28800
Large	3	Medium	High	30-70%	1392.1	3517.2	28800
Large	3	Medium	High	50-50%	2350	5291.5	28800
Large	3	Medium	High	70-30%	3258.1	7630	28800
Large	3	Medium	Low	30-70%	1419.5	3858.9	28800
Large	3	Medium	Low	50-50%	2361.7	5858	28800
Large	3	Medium	Low	70-30%	3300	8275.4	28800
Large	3	Medium	Medium	30-70%	1171.2	2772	28800
Large	3	Medium	Medium	50-50%	1936.2	6002	28800
Large	3	Medium	Medium	70-30%	2723.9	9411.3	28800

Table D.2 LBBD results in large examples

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	LBBD		
					LB	UB	Time
Large	1	High	High	30-70%	1433.4		28800
Large	1	High	High	50-50%	2397		28800
Large	1	High	High	70-30%	3308.2		28800
Large	1	High	Low	30-70%	3384	4002.1	28800
Large	1	High	Low	50-50%	3567.5	3804.5	28800
Large	1	High	Low	70-30%	4988.2		28800
Large	1	High	Medium	30-70%	1474.5		28800
Large	1	High	Medium	50-50%	2530		28800
Large	1	High	Medium	70-30%	3448.9		28800
Large	1	Low	High	30-70%	1140.9		28800
Large	1	Low	High	50-50%	1900	3052	28800
Large	1	Low	High	70-30%	2660		28800
Large	1	Low	Low	30-70%	2734.2	5419.5	28800
Large	1	Low	Low	50-50%	3940	6208	28800
Large	1	Low	Low	70-30%	5091.8	6643.3	28800
Large	1	Low	Medium	30-70%	1200.6		28800
Large	1	Low	Medium	50-50%	1772		28800
Large	1	Low	Medium	70-30%	2335.1	3759.7	28800
Large	1	Medium	High	30-70%	1442.4		28800
Large	1	Medium	High	50-50%	2404		28800
Large	1	Medium	High	70-30%	3365.6		28800
Large	1	Medium	Low	30-70%	3982	4140.1	28800
Large	1	Medium	Low	50-50%	5440.5	5473.5	28800
Large	1	Medium	Low	70-30%	6056.4	6600.4	28800
Large	1	Medium	Medium	30-70%	1807.8		28800
Large	1	Medium	Medium	50-50%	2991.5	4278	28800
Large	1	Medium	Medium	70-30%	4205.6		28800
Large	2	High	High	30-70%	1428.9		28800
Large	2	High	High	50-50%	2376.5		28800
Large	2	High	High	70-30%	3388.7		28800
Large	2	High	Low	30-70%	2086.1		28800
Large	2	High	Low	50-50%	3116		28800
Large	2	High	Low	70-30%	4227.5		28800
Large	2	High	Medium	30-70%	1608.6		28800
Large	2	High	Medium	50-50%	2681		28800
Large	2	High	Medium	70-30%	3753.4		28800
Large	2	Low	High	30-70%	1516.8		28800
Large	2	Low	High	50-50%	2520.5		28800
Large	2	Low	High	70-30%	3537.1		28800
Large	2	Low	Low	30-70%	2437.5		28800
Large	2	Low	Low	50-50%	3949.5		28800
Large	2	Low	Low	70-30%	4437.8		28800

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	LBBD		
					LB	UB	Time
Large	2	Low	Medium	30-70%	1608.6		28800
Large	2	Low	Medium	50-50%	3602		28800
Large	2	Low	Medium	70-30%	5355.7		28800
Large	2	Medium	High	30-70%	2110.2		28800
Large	2	Medium	High	50-50%	3425.5		28800
Large	2	Medium	High	70-30%	4838.5		28800
Large	2	Medium	Low	30-70%	4374.7		28800
Large	2	Medium	Low	50-50%	5054		28800
Large	2	Medium	Low	70-30%	5074.8		28800
Large	2	Medium	Medium	30-70%	1230.3		28800
Large	2	Medium	Medium	50-50%	2050.5		28800
Large	2	Medium	Medium	70-30%	2870.7		28800
Large	3	High	High	30-70%	2441.7		28800
Large	3	High	High	50-50%	3972		28800
Large	3	High	High	70-30%	5569.9		28800
Large	3	High	Low	30-70%	2376.7		28800
Large	3	High	Low	50-50%	3489.5		28800
Large	3	High	Low	70-30%	4537		28800
Large	3	High	Medium	30-70%	1916.4		28800
Large	3	High	Medium	50-50%	3327		28800
Large	3	High	Medium	70-30%	4541.6		28800
Large	3	Low	High	30-70%	2176.5		28800
Large	3	Low	High	50-50%	3707.5		28800
Large	3	Low	High	70-30%	5078.5		28800
Large	3	Low	Low	30-70%	2700.8		28800
Large	3	Low	Low	50-50%	4412.5		28800
Large	3	Low	Low	70-30%	5524.3		28800
Large	3	Low	Medium	30-70%	1847.7		28800
Large	3	Low	Medium	50-50%	3197.5		28800
Large	3	Low	Medium	70-30%	4189.5		28800
Large	3	Medium	High	30-70%	1866.6		28800
Large	3	Medium	High	50-50%	3068.5		28800
Large	3	Medium	High	70-30%	4478.6		28800
Large	3	Medium	Low	30-70%	1921.3		28800
Large	3	Medium	Low	50-50%	3417.5		28800
Large	3	Medium	Low	70-30%	4370.7		28800
Large	3	Medium	Medium	30-70%	1670.2		28800
Large	3	Medium	Medium	50-50%	2911.1		28800
Large	3	Medium	Medium	70-30%	3987.9		28800

Table D.3 F-LPR and P-LPR results in large examples

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	F-LPR		P-LPR	
					LB	Time	LB	Time
Large	1	High	High	30-70%	1206	28800	1196.7	28800
Large	1	High	High	50-50%	1946.5	28800	1981.5	28800
Large	1	High	High	70-30%	2741.2	28800	2693.4	28800
Large	1	High	Low	30-70%	3413	28800	3400	28800
Large	1	High	Low	50-50%	3567.5	28800	3567.5	28800
Large	1	High	Low	70-30%	5144	28800	4990.7	28800
Large	1	High	Medium	30-70%	1292.1	28800	1456.8	28800
Large	1	High	Medium	50-50%	2153.5	28800	2448	28800
Large	1	High	Medium	70-30%	3166.8	28800	3306.5	28800
Large	1	Low	High	30-70%	1168.2	28800	1193.7	28800
Large	1	Low	High	50-50%	1464.7	28800	1464.7	28800
Large	1	Low	High	70-30%	2050.5	28800	2050.5	28800
Large	1	Low	Low	30-70%	2742.2	28800	2817.7	28800
Large	1	Low	Low	50-50%	4799.4	28800	3811.9	28800
Large	1	Low	Low	70-30%	5018.8	28800	4953.9	28800
Large	1	Low	Medium	30-70%	1152.3	28800	1215.2	28800
Large	1	Low	Medium	50-50%	1795.1	28800	1871	28800
Large	1	Low	Medium	70-30%	2287.7	28800	2260.6	28800
Large	1	Medium	High	30-70%	1304.1	28800	1291.2	28800
Large	1	Medium	High	50-50%	2178	28800	2024.6	28800
Large	1	Medium	High	70-30%	3087	28800	2838.5	28800
Large	1	Medium	Low	30-70%	4021.7	28800	4021.7	28800
Large	1	Medium	Low	50-50%	5473.5	28800	5473.5	28800
Large	1	Medium	Low	70-30%	4437.8	28800	4758.6	28800
Large	1	Medium	Medium	30-70%	1738.8	28800	1656	28800
Large	1	Medium	Medium	50-50%	2810	28800	2817.8	28800
Large	1	Medium	Medium	70-30%	3806.2	28800	3791.4	28800
Large	2	High	High	30-70%	1326.6	28800	1253.4	28800
Large	2	High	High	50-50%	2205.5	28800	2046	28800
Large	2	High	High	70-30%	3033.8	28800	2922.5	28800
Large	2	High	Low	30-70%	2085.2	28800	1275.9	28800
Large	2	High	Low	50-50%	3157	28800	3141	28800
Large	2	High	Low	70-30%	3247.2	28800	4138.4	28800
Large	2	High	Medium	30-70%	1422.1	28800	1148.5	28800
Large	2	High	Medium	50-50%	2527.8	28800	2272	28800
Large	2	High	Medium	70-30%	3151.1	28800	3146.6	28800
Large	2	Low	High	30-70%	1781.5	28800	1710.9	28800
Large	2	Low	High	50-50%	3080.8	28800	3061.2	28800
Large	2	Low	High	70-30%	4023.3	28800	4060.8	28800
Large	2	Low	Low	30-70%	1776.1	28800	1776.1	28800
Large	2	Low	Low	50-50%	2813.5	28800	2806.5	28800
Large	2	Low	Low	70-30%	3393.6	28800	3425.2	28800

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	F-LPR		P-LPR	
					LB	Time	LB	Time
Large	2	Low	Medium	30-70%	1405.4	28800	1617.8	28800
Large	2	Low	Medium	50-50%	3428.4	28800	3333.4	28800
Large	2	Low	Medium	70-30%	4708	28800	4629.3	28800
Large	2	Medium	High	30-70%	2302.2	28800	1593	28800
Large	2	Medium	High	50-50%	3855	28800	2631.3	28800
Large	2	Medium	High	70-30%	4903.8	28800	3719.1	28800
Large	2	Medium	Low	30-70%	4464.6	28800	4457.7	8902
Large	2	Medium	Low	50-50%	4906.5	28800	5114	4413
Large	2	Medium	Low	70-30%	5151	28800	5133	5908
Large	2	Medium	Medium	30-70%	1249.2	28800	1103.1	28800
Large	2	Medium	Medium	50-50%	2106.2	28800	1837	28800
Large	2	Medium	Medium	70-30%	2940	28800	2597.7	28800
Large	3	High	High	30-70%	1804	28800	1586.4	28800
Large	3	High	High	50-50%	3020.9	28800	2988	28800
Large	3	High	High	70-30%	4128.1	28800	3694.7	28800
Large	3	High	Low	30-70%	2381.8	28800	2363.2	28800
Large	3	High	Low	50-50%	3467	11982	3467	7968
Large	3	High	Low	70-30%	4505.5	28800	4505.5	28800
Large	3	High	Medium	30-70%	1563.6	28800	1336	28800
Large	3	High	Medium	50-50%	2500.5	28800	2226.6	28800
Large	3	High	Medium	70-30%	3589.1	28800	3099	28800
Large	3	Low	High	30-70%	2307.1	28800	2312	28800
Large	3	Low	High	50-50%	3950.8	28800	3977.4	28800
Large	3	Low	High	70-30%	5449.4	28800	5421.4	28800
Large	3	Low	Low	30-70%	2754.4	28800	2411.2	28800
Large	3	Low	Low	50-50%	3931.9	28800	3919.7	28800
Large	3	Low	Low	70-30%	5245.3	28800	5128.9	28800
Large	3	Low	Medium	30-70%	2254.1	28800	2300.8	28800
Large	3	Low	Medium	50-50%	3800	28800	3896	28800
Large	3	Low	Medium	70-30%	5211.8	28800	5303	28800
Large	3	Medium	High	30-70%	1945.5	28800	1712.6	28800
Large	3	Medium	High	50-50%	3087.5	28800	2854.6	28800
Large	3	Medium	High	70-30%	4319	28800	4536.9	28800
Large	3	Medium	Low	30-70%	1338	2	1338	5
Large	3	Medium	Low	50-50%	2668.2	28800	2230	3
Large	3	Medium	Low	70-30%	3122	2	3122	4
Large	3	Medium	Medium	30-70%	1196.2	28800	1134.1	28800
Large	3	Medium	Medium	50-50%	2042.7	28800	2374	28800
Large	3	Medium	Medium	70-30%	2541.2	28800	2682.5	28800

Table D.4 Search algorithm results in large examples

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	TS-CL		TS-IL	
					UB	Time	UB	Time
Large	1	High	High	30-70%	1786.2	8519	1839	560
Large	1	High	High	50-50%	2801.5	26933	3049	1159
Large	1	High	High	70-30%	4294.5	4728	4504.5	936
Large	1	High	Low	30-70%	3933.4	410	3818.5	228
Large	1	High	Low	50-50%	4153	367	3734	543
Large	1	High	Low	70-30%	5105.7	700	5253.4	587
Large	1	High	Medium	30-70%	1929.6	4254	2077.8	460
Large	1	High	Medium	50-50%	3154	13627	3305	1546
Large	1	High	Medium	70-30%	4253.5	16481	5418.7	427
Large	1	Low	High	30-70%	1932.9	5337	2087.4	430
Large	1	Low	High	50-50%	2906	6056	3072	456
Large	1	Low	High	70-30%	4492.4	648	4492.4	258
Large	1	Low	Low	30-70%	4320.4	1640	4566.4	110
Large	1	Low	Low	50-50%	5051.5	3759	5695.5	169
Large	1	Low	Low	70-30%	6910.8	2061	7191.6	91
Large	1	Low	Medium	30-70%	2115.7	1486	2257.7	224
Large	1	Low	Medium	50-50%	3183	4633	3498.5	548
Large	1	Low	Medium	70-30%	3977.9	1192	4029	399
Large	1	Medium	High	30-70%	2283.8	1054	2206.1	743
Large	1	Medium	High	50-50%	4021	1800	4021	1056
Large	1	Medium	High	70-30%	4971.9	973	5064	531
Large	1	Medium	Low	30-70%	4221.9	1579	4339.7	143
Large	1	Medium	Low	50-50%	5685.5	2665	6100.5	301
Large	1	Medium	Low	70-30%	6540.4	2328	6663.4	198
Large	1	Medium	Medium	30-70%	2624.8	14004	3217	723
Large	1	Medium	Medium	50-50%	3867	18185	4955	548
Large	1	Medium	Medium	70-30%	5630.5	12527	6667.6	404
Large	2	High	High	30-70%	1814.4	2787.5	2415	322
Large	2	High	High	50-50%	3023.5	27994	3343.5	1630
Large	2	High	High	70-30%	4379.2	28800	5100.2	695
Large	2	High	Low	30-70%	2268.4	351	2268.4	45
Large	2	High	Low	50-50%	3526.5	1723	3912.5	169
Large	2	High	Low	70-30%	4320.5	1496	4766.3	137
Large	2	High	Medium	30-70%	2117.1	15180	2500.2	615
Large	2	High	Medium	50-50%	3476.5	15593	4397.5	1901
Large	2	High	Medium	70-30%	4939.2	18804	5895.5	559
Large	2	Low	High	30-70%	2645.4	456	2573.3	225
Large	2	Low	High	50-50%	3771.5	1736	3973	2296
Large	2	Low	High	70-30%	6020.2	482	6150.1	1969
Large	2	Low	Low	30-70%	3453.6	2232	3453.6	1129
Large	2	Low	Low	50-50%	4842.5	6004	5141.5	1509
Large	2	Low	Low	70-30%	5568.5	3962	6715.5	992

Size	Example	Labor Flexibility	Machine Flexibility	Scenario	TS-CL		TS-IL	
					UB	Time	UB	Time
Large	2	Low	Medium	30-70%	2097.9	18400	2500.2	778
Large	2	Low	Medium	50-50%	5773.5	14341	5849	1797
Large	2	Low	Medium	70-30%	8269.9	11318	8908.3	691
Large	2	Medium	High	30-70%	3204.2	28800	3674.3	1077
Large	2	Medium	High	50-50%	5819.5	28800	6076.5	9171
Large	2	Medium	High	70-30%	7877.7	28800	9416	890
Large	2	Medium	Low	30-70%	5074.7	981	5422.1	88
Large	2	Medium	Low	50-50%	5645	1964	6598	129
Large	2	Medium	Low	70-30%	5960	1650	7867.2	101
Large	2	Medium	Medium	30-70%	1676.6	3944	1881.6	149
Large	2	Medium	Medium	50-50%	2465	6905	2564	246
Large	2	Medium	Medium	70-30%	3547.1	6617	3684.7	277
Large	3	High	High	30-70%	4079.1	8521	4179.6	943
Large	3	High	High	50-50%	6833	28800	7754	1384
Large	3	High	High	70-30%	9905.7	7700	9858.8	5602
Large	3	High	Low	30-70%	2518.6	269	2518.6	35
Large	3	High	Low	50-50%	3574.5	2142	3761	25
Large	3	High	Low	70-30%	4707.6	2142	5088.6	54
Large	3	High	Medium	30-70%	2814.6	19925	3120.6	3690
Large	3	High	Medium	50-50%	4442.5	16110	4893	6382
Large	3	High	Medium	70-30%	7140	16515	7591.5	5431
Large	3	Low	High	30-70%	5227.2	15404	5316.8	781
Large	3	Low	High	50-50%	7305	28800	7712	3052
Large	3	Low	High	70-30%	11293.5	28800	11643.6	1843
Large	3	Low	Low	30-70%	3125.1	17092	3274.2	624
Large	3	Low	Low	50-50%	5688	28800	5917	844
Large	3	Low	Low	70-30%	7097.6	17946	7097.6	4280
Large	3	Low	Medium	30-70%	4115.8	3944	4304.8	844
Large	3	Low	Medium	50-50%	6046	10318	6131.5	1932
Large	3	Low	Medium	70-30%	8088.9	11406	8088.9	3757
Large	3	Medium	High	30-70%	3514.6	9670	3529	1023
Large	3	Medium	High	50-50%	5165	21378	5653	10884
Large	3	Medium	High	70-30%	7406	28800	8458.5	7573
Large	3	Medium	Low	30-70%	3277.2	28800	3653.1	1703
Large	3	Medium	Low	50-50%	4451	28800	5863.5	3351
Large	3	Medium	Low	70-30%	6835.1	28800	7728.1	4335
Large	3	Medium	Medium	30-70%	2892.8	14253	3539.8	838
Large	3	Medium	Medium	50-50%	5640.5	28800	7565	1256
Large	3	Medium	Medium	70-30%	8524.6	28800	10014.9	1755