INTEGRATED SCHEDULING OF PRODUCTION AND LOGISTICS OPERATIONS OF A MULTI-PLANT MANUFACTURER SERVING A SINGLE CUSTOMER AREA

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By
Merve Çelen
July, 2009

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

_____

Asst. Prof. Dr. Mehmet Rüştü Taner (Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

_____

Assoc. Prof. Dr. Oya Ekin Karaşan

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

_____

Asst. Prof. Dr. Sedef Meral

Approved for the Institute of Engineering and Science

_____

Prof. Dr. Mehmet Baray
Director of Institute of Engineering and Science

ABSTRACT

INTEGRATED SCHEDULING OF PRODUCTION AND LOGISTICS OPERATIONS OF A MULTI-PLANT MANUFACTURER SERVING A SINGLE CUSTOMER AREA

Merve Çelen

M.S. in Industrial Engineering

Supervisor: Asst. Prof. Dr. Mehmet Rüştü Taner

July, 2009

Increasing market competition forces manufacturers to continuously reduce their leadtimes by minimizing the total time spent in both production and distribution. Some studies in the relevant literature indicate that scheduling production and logistics operations in a coordinated manner leads to improved results. This thesis studies the problem of scheduling production and distribution operations of a manufacturer serving a single customer area from multiple identical production plants dispersed at different geographical locations. The products are transported to the customer area by a single capacitated truck. The setting is inspired by the operations of a leading soft drink manufacturer, and the objective is set in line with their needs as the minimization of the total completion time of the jobs. The completion time of a job is defined as the time it reaches at the customer area. We consider both this general problem and four special cases motivated by common practical applications. We prove that both the main problem and three of its special cases are NP-hard at least in the ordinary sense. We develop mixed integer programming (MIP) models for all these problems and propose a pseudo-polynomial dynamic programming mechanism for the remaining special case. Since the MIP models are able to provide optimal solutions only for small instances in a reasonable amount of time, heuristics are also proposed to solve larger instances. Fast lower bounds are developed to facilitate the performance assessment of these heuristics in medium and large instances. Evidence from extensive computational experimentation suggests that the proposed heuristics are both efficient and effective.

Keywords: Multi-plant scheduling, distribution, complexity, integer programming, heuristic

ÖZET

TEK MÜŞTERİ BÖLGESİNE HİZMET VEREN ÇOK TESİSLİ BİR İMALATÇININ ÜRETİM ÇİZELGELEME VE LOJİSTİK FAALİYETLERİNİN BÜTÜNLEŞİK PLANLAMASI

Merve Çelen
Endüstri Mühendisliği, Yüksek Lisans
Tez Yöneticisi: Yrd. Doç. Dr. Mehmet Rüştü Taner
Temmuz, 2009

Yükselen pazar rekabeti imalatçıları üretim ve dağıtımda geçen toplam süreyi en küçülterek tedarik sürelerini sürekli olarak azaltmaya zorlamaktadır. İlgili literatürdeki bazı çalışmalar üretim ve dağıtım işlemlerinin bağlantılı bir şekilde çizelgelenmesinin daha iyi sonuçlar verdiğini göstermektedir. Bu tezde farklı coğrafik yerseçimlerindeki özdeş tesislerden tek müşteri bölgesine hizmet veren bir imalatçının üretim ve dağıtım işlemlerinin çizelgelenmesi çalışılmaktadır. Ürünler müşteri bölgesine taşıma kapasitesi belirli bir araç tarafından taşınmaktadır. Problemin tanımı ileri gelen bir alkolsüz içecek imalatçısının çalışmalarından esinlenilmiş ve amaç fonksiyonu onların ihtiyaçlarıyla uyumlu olacak şekilde işlerin bitiş zamanlarının toplamının en küçültülmesi olarak belirlenmiştir. Bir işin bitiş zamanı o işin müşteri bölgesine ulaştığı zaman olarak tanımlanmıştır. Hem genel problem hem de onun pratik uygulamalarda sıklıkla karşılaşılan dört özel durumu ele alınmıştır. Hem esas problemin hem de onun üç özel durumunun NP-zor olduğu kanıtlanmıştır. Tüm bu problemler için karışık tamsayılı programlama (KTP) modelleri geliştirilmiş ve geri kalan özel durum için sözde-polinom bir dinamik programlama mekanizması önerilmiştir. KTP modelleri eniyi sonuçları makul sürelerde sadece küçük örnekler için verebildiğinden büyük örnekleri çözebilmek için sezgisel yöntemler önerilmiştir. Sezgisel yöntemlerin orta ve büyük örneklerdeki performans değerlendirmesini kolaylaştırmak amacıyla hızlı alt sınırlar

geliştirilmiştir. Yapılan kapsamlı deneyler önerilen sezgisel yöntemlerin verimli ve etkili olduğunu göstermiştir.

Anahtar sözcükler: Çok tesisli çizelgeleme, dağıtım, karmaşıklık, tamsayı programlama, sezgisel yöntemler

To my mother...

Acknowledgement

First, I would like to express profound gratitude to my advisor, Asst. Prof. Dr. Mehmet Rüştü Taner, for sharing his experience and knowledge with me. I would like to thank him especially for tolerating me when I was upset and nervous.

I am also grateful to Assoc. Prof. Dr. Oya Ekin Karaşan and Asst. Prof. Dr. Sedef Meral for accepting to read and review this thesis and for their invaluable suggestions.

I am as ever, especially indebted to my family, Tülün, Ender and Elif Çelen, for their endless love and support. Everyday on the phone they tried to calm me down and motivate me for hours. Whenever I had a break down, their motivations made me continue. Even when we were at different continents their love was always with me and I would not be able to finish this thesis without them.

I want to thank my roommates Hande Koçak and Çiğdem Ataseven for sharing the life with me for two years.

My sincere thanks go to my friends Bora Karaoğlu, Büşra Çelikkaya, Hayrettin Gürkök, Tuncer Erdoğan, Adnan Tula, Utku Guruşçu, İhsan Yanıkoğlu, Safa Onur Bingöl, Emre Uzun, Hatice Çalık, Sıtkı Gülten, Serdar Yıldız, İpek Şener, Sezgin Küçükçoban and Erkin Bahçeci.

Finally, I would like to express my special thanks to TÜBİTAK for the scholarship provided throughout this thesis study.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

The main focus of scheduling is the allocation of limited resources to tasks over time with the objective of optimizing with respect to one or more performance measures. Recent developments in scheduling have focused on extending various classical models to address real-life problems more closely. In today's competitive business world, as the concept of on-time delivery of jobs has become more crucial for customer satisfaction, effective scheduling of production resources, achieving reduction in inventory levels and shortening lead times have gained much criticality.

Many studies in scheduling literature have concentrated on how to effectively schedule production operations within the confines of a single production facility. However, from the perspective of minimizing the total cost in a supply chain, companies usually acknowledge that the cost of a product is not only determined with the amount of factory resources used to convert the raw material into a finished product, but also with the amount of resources used to deliver the product to the customer. Hence, concentrating only on scheduling of production operations within plants may not be sufficient to obtain the desired low levels in the production and logistics costs of the supply chain. From another perspective to increase customer satisfaction, the order lead times should also be minimized. This requies the time spent both in the production of the product and in its distribution to be minimized

The importance of the coordination between production and distribution operations have been studied in Chandra and Fisher (1994), Ertogral et al. (1998), and Fumero and

Vercellis (1999). It has been shown that integrated scheduling of production and distribution operations perform substantially better than unsynchronized scheduling of these operations. Hence, it is important for the companies to recognize that a reduction in total cost of the supply chain and an increase in customer satisfaction can be realized through integrated scheduling of production and distribution operations.

In the relevant literature there have been various studies that concentrate on the integration of production and distribution operations. However, most of those studies have considered a single production plant with different machine configurations. To the best of our knowledge, there are only two studies in the relevant literature that try to address the problem with decentralized plant locations, the studies of Chen and Pundoor (2006) and Li and Ou (2007). In these works, the products have a very short selling season, and are transported from the plants to a warehouse via a third-party carrier and it is assumed that a transporter would be available at each facility whenever one is required. However, as experienced by a leading soft drink manufacturer in Turkey, third-party carriers may cause high distribution costs and long lead times. Because of increasing demand the prices of the trucks supplied by these third-party carriers are increasing while the availability of the trucks is decreasing. Therefore, aforementioned soft drink manufacturer has considered building its own fleet.

In order to address the problems faced by the soft drink manufacturer, in this thesis, we study the problem of integrated scheduling of production and distribution operations of a manufacturer with multiple production plants at different locations serving a single customer area via a single capacitated truck. Assuming that a truck is assigned to each warehouse and that truck is used to transport the products from the plants to that warehouse, our work can also be used to find a good solution for a larger system with decentralized plants and multiple warehouses.

As in the problem studied in Qi (2008), in order to satisfy unexpected customer demand on time, some companies may choose outsourcing from other plants at different

locations. Assuming that the outsourcing company has a single capacitated truck, proposed methods in this thesis can be used to address the problem of outsourcing.

The rest of this thesis is organized as follows. The next chapter gives a summary of the relevant literature in the area of integrated scheduling of production and distribution operations. Chapter 3 gives a precise explanation of the problem setting and necessary notations. In Chapter 4, dynamic programming algorithms for the problem where the assignments of the jobs to the plants are known are provided. In addition to that, heuristic and exact algorithms and lower bound generation methods for the three special cases and the original problem are explained in detail. Following that, in Chapter 5, the experimental design to test the performance of the proposed methods and the numerical results obtained in the generated test problems are given. A general conclusion of the study is presented in Chapter 6.

Chapter 2

LITERATURE REVIEW

There are many studies in the literature which consider the integration of production and distribution operations. We study the problem with *m* identical plants at different locations serving a single customer area. We consider a single truck with finite capacity to transport the jobs from the plants to the customer area. To give the related literature, we will first review the papers according to their fleet types in the Section 2.1. Then in Section 2.2, we will continue with the review of the literature based on the machine configuration and customer area properties.

2.1. Studies with Different Fleet Types

The fleet types considered in the scheduling literature can mainly be grouped based on two aspects: the number of trucks in the fleet and the capacity of the trucks. Most of the studies up to date assume a fleet consisting of infinitely many trucks, a few of which integrate a capacity constraint for the trucks.

The first papers in scheduling research that focuses on problems where the completion time of a job is defined as the time the job reaches the customer, mainly consider a fleet of infinitely many trucks. They do not consider a capacity constraint for the trucks. Although this assumption of infinite capacity simplifies the problem, it is not realistic. Potts (1980), Hall and Shmoys (1989), Zdrzalka (1991, 1995) and Woeginger (1994, 1998) are the

first ones to use this assumption. They assume that whenever a job finishes its production, there is always a truck available to immediately deliver the job to the customer in q_j time where q_j denotes the time required to transport job *j* from the machine/plant to the customer. Hence the time a job reaches the customer is the process completion time of that job at the plant plus the transportation time of that job.

In the studies of Cheng et al. (1996), Cheng and Gordon (1994), Cheng et al. (1997) and Wang and Cheng (2000), the delivery time of a batch is defined as the completion time of the last job in that batch. Here, the transportation is assumed to be instantaneous. However, the objective functions to minimize contain a term for delivery cost and delivery cost depends on the number of deliveries made to transport all the jobs from the plant to the customer. Hall and Potts (2003) make the same instantaneous transportation assumption in their paper that studies scheduling problems in a supply chain where a supplier makes deliveries to several manufacturers. Also these manufacturers make deliveries to customers. Again in the study of Qi (2006), the transportation from the facility to the customer is assumed to be instantaneous. However, for the transportation between facilities at different locations, infinitely many trucks are used and a transshipment time is required.

The study of Hall and Potts (2005) also investigates the condition where there are infinitely many trucks. However, they also consider the case when there is only one truck. They combine these two different truck constraints by defining a parameter *T* which denotes the minimum time between any two consecutive deliveries. If there are a sufficient number of trucks, *T* is defined as the time it takes to load a truck; else if there is a single truck, it is defined as the travel time to and from the customers plus the loading time.

The papers of Pundoor and Chen (2005) and Chen and Vairaktarakis (2005) consider the case with a fleet of infinitely many trucks. However, their fleet choice is more realistic when compared with the aforementioned ones since they take the capacity constraint into account. In Chen and Vairaktarakis (2005), it is assumed that distribution operations are

carried out by third-party carriers and thus an unlimited number of vehicles are available. But due to limited vehicle capacity, each truck can carry up to a certain number of jobs. The same third-party carrier assumption with vehicle capacities also holds in the study of Li and Vairaktarakis (2007).

Chang and Lee (2004) study a more difficult problem by considering a single and capacitated truck. They define the completion time of a sequence as the time when the vehicle finishes delivering the last batch to the customer site(s) and returns to the machine(s). The reason for this definition is that to have a cyclic pattern they have to wait for the truck to return to the machine. An interesting point to mention about their study is that they assume each job might occupy a different amount of physical space in the truck. Li et al. (2005) also incorporate the same single and capacitated truck idea in their study. However, their capacity constraint is defined so that the truck can also behave as uncapacitated. They define the capacity of the truck to be K and they add that K can also be equal to the number of jobs, i.e. n, which means an unlimited capacity.

Although it may seem that third-party carrier assumption is reasonable, it should also be taken into account that to decrease transportation cost, many manufacturing companies prefer to build their own fleet. Since the number of trucks in such a fleet will be limited, the assumption of infinitely many trucks becomes unreasonable. It is also for sure that for a more realistic assumption, each truck will have a capacity constraint. From this perspective, Lee and Chen (2001) can be considered as a milestone in the literature relevant to the integration of production and distribution. They study two classes of transportation: within the facility and from the facility to the customer. For both transportation classes, they consider various types of fleet. The types of fleet they assume in their study can be listed as follows:

- Single truck which can carry only one job at a time
- Single truck which can carry more than one but fewer than n jobs
- Single truck which can carry $n/2$ jobs

- Multiple but limited number of trucks, each can carry more than one but fewer than *n* jobs

Although single truck case has been studied before, Lee and Chen (2001) is the first to consider multiple trucks.

As mentioned in Chapter 1 for the soft drink manufacturer, using third-party carriers may cause high distribution costs and long leadtimes. To reduce the distribution costs and leadtimes, it is a reasonable alternative for the companies to have their own fleet instead of using third-party carriers. It is logical for large companies to assign a truck to each of their warehouses and each warehouse would be responsible from using its own truck to have their products delivered to them from the plants. In this context, in our problem, having a single truck with a finite capacity is realistic. Also, for a small company which uses outsourcing from plants at different locations, having a single capacitated truck to gather the products from other plants is logical.

2.2. Studies with Different Machine Configuration and Customer Area Properties

In the literature relevant to simultaneous scheduling of production and distribution, there have been various studies considering different types of machine configurations and different structures for the customer location, e.g.. a single customer area or multiple customer areas. For the machine configuration part, there are many papers that study single machine, parallel machines or a series of flow shop machines. So it would be beneficial to group the relevant literature according to their combinations of machine configuration and customer area. In such a case, four main groups may occur:

- Single machine and single customer area

- Single machine and multiple customer areas
- Multiple machines and single customer area
- Multiple machines and multiple customer areas

We will review the studies in each group in the same order as they are stated above.

2.2.1. Single Machine and Single Customer Area

Most of the papers, especially the earliest ones, in the relevant literature consider this case in their studies. The reason for this is that single machine scheduling is relatively easier than multiple machine scheduling. Also sending the jobs to a single customer area eliminates routing issues and hence the problem becomes easier. Some of the papers in this area consider batching, which means that each batch will consist of a pre-specified amount of jobs. We will first review the studies which do not consider a batch concept as defined above.

The first paper that considers the scheduling of the production on a single machine and delivery of the finished products to a single customer is the study of Potts (1980). In his paper, Potts tries to find a sequence for the jobs, each of which has a release date (r_i), a nonnegative processing time (p_i) and a delivery time (q_i). The objective is to minimize the time by which all jobs are delivered. One of the interesting points in this study is the *symmetric form* of the original problem created by defining due dates for each job *i* by assigning $d_i = K - q_i$ where *K* is a constant. This forms a modified problem in which due dates replace the delivery times. Hence, minimizing the makespan in the symmetric form is equivalent to minimizing maximum lateness with respect to the due dates. This problem was shown to be NP-hard by Lenstra et al. (1977) and Potts develops a heuristic that is guaranteed to produce a solution within 50% of the optimum.

Hall and Shmoys (1989) also consider release times, processing times and delivery times for the jobs. With a similar modification of the problem as in Potts (1980), they convert the makespan problem into a maximum lateness problem. They denote the problem as $1|r_j|L_{\max}$ using the notation of Graham et al. (1979) and they provide a polynomial time approximation scheme for this problem. For the special case, where r_j = 0, they use the Jackson's Rule to solve the problem optimally in polynomial time. They also investigate the case where precedence constraints between jobs are involved. They provide a (4/3)-approximation algorithm for $1|r_j, prec|L_{\max}$. Hall and Shmoys (1992) give a detailed explanation of their (4/3)-approximation scheme and provide two polynomial approximation schemes for the problem $1|r_j|L_{\max}$. Woeginger (1994) considers similar job properties but with the exception that he does not take release times into account. For the single machine case, Woeginger modifies the problem as in Potts (1980) and transforms it into $1||L_{\max}$. It is stated that sequencing the jobs in non-increasing delivery times, i.e. sequencing the jobs according to *earliest due date* rule, yields an optimum schedule in $O(n \log n)$ time. The author also claims that no algorithm that is asymptotically faster than $O(n \log n)$ can construct an optimum schedule for this problem.

Lee and Chen (2001) consider two different objectives in their study: minimizing the time by which the last job reaches the customer and minimizing the total flow time. The main importance of their study is that they assume a limited number of trucks with finite capacity and hence there is not a delivery time assigned to each job. In fact the delivery time for each job is determined by the distance of the machine to the customer. For the objective of minimizing the makespan, the paper shows that this single machine problem with multiple trucks can easily be converted to a parallel machine problem where each truck is treated as a machine. The modified problem is denoted as $Pv|r_j, p_j \equiv p|C_{\max}$ where v is the number of trucks, p is the time to travel from the machine to the customer and turn back to the machine. It is stated that this problem can be solved optimally in polynomial time. For the total flow

time case, a polynomial time dynamic programming algorithm is presented. Later Li et al. (2005) present a more efficient dynamic programming algorithm for this case. Chang and Lee (2004) consider a similar problem but with a single truck condition. It is shown that minimizing the makespan subject to these constraints is NP-hard in the strong sense. The authors provide a heuristic for the problem which produces solutions with an error bound of 5/3.

Hall and Potts (2005) again study the single plant and single customer area setting with finitely many capacitated trucks, but also in order to minimize the number of trips required to deliver all the jobs, the authors include the transportation cost into their objective functions. Their study considers a constraint *T*, which denotes the minimum time interval between two consecutive deliveries which was explained in more detail in Section 2.1. For the problems $1|T|Dy+\sum C_j$, $1|T|Dy+L_{\max}$, and $1|T|Dy+\sum U_j$, where *D* denotes the delivery cost for each trip and *y* denotes the number of trips, polynomial time algorithms are provided to find an optimal schedule whereas for the problems $1|T|Dy+\sum w_jU_j$ and $1|\ |Dy+\sum T_j$ pseudopolynomial algorithms are presented. The study also shows that recognition versions of problems $1|\ |Dy+\sum w_jC_j$ and $1|T|Dy+\sum w_jC_j$ are unary NP-complete. Chen and Vairaktarakis (2005) also incorporate distribution cost into their objective functions but differently from Hall and Potts (2005) infinitely many capacitated trucks are considered. The objective is to minimize total distribution cost and mean (or maximum) delivery time simultaneously and polynomial time algorithms are provided for these problems.

In Li and Ou's (2005) work, a single capacitated truck is used to carry unprocessed jobs from the warehouse to the plant and processed jobs from the plant to the warehouse. The study aims to minimize the makespan and polynomial time algorithms for some special cases and a heuristic for the general problem are proposed.

Although Qi (2006) considers two machines at different locations, since each machine serves to its own customer area, it is more suitable to review this study in this section. The reason for the authors to mention the locations of the plants is that whenever the demand in one of the processing facilities, named *facility 1*, exceeds its capacity the other facility's, named *facility 2*, capacity may be used and the jobs processed there should be transferred to *facility 1* again. Then the jobs are delivered from *facility 1* to its customer. Both batch transshipment and item transshipment are considered under different objective functions and algorithms (mostly dynamic programming) are developed for them.

The notion of batching starts with Zdrzalka (1991). In this study, a unit setup time associated with switching from jobs in one batch to those in another is inserted. Three polynomial time heuristics so as to minimize the time by which the last batch reaches the customer are proposed. Zdrzalka (1995) generalizes the problem to sequence independent setup times and provides two approximation algorithms with the worst-case performance ratio of 3/2. In Woeginger (1998), a polynomial-time approximation scheme is provided for the problem in Zdrzalka (1995). In addition to that, it is shown that finding a polynomial-time approximation algorithm with constant worst-case ratio is not possible for a variant of the problem where sequence-dependent setup times are involved. Setup times are involved also in the study of Cheng et al. (1997). The authors assume that there is a constant setup time between two consecutive batches. The objective is to find a number B of batches and a sequence so as to minimize the sum of the total weighted job earliness and mean batch delivery time. They prove that the problem is NP-hard in the strong sense. It is stated that even for the case where the number of batches has a fixed upper bound, the problem is NP-hard in the ordinary sense and a dynamic programming algorithm which solves this problem in pseudopolynomial time is provided. For this upper bound case, it is shown that the complexity does not change even if the setup times are equal to zero. Also the authors prove when B has a fixed lower bound the problem remains strongly NP-hard. For the special cases where all the weights are equal or all the processing times are equal, polynomial time algorithms are derived. Finally, a heuristic approach is presented for the general problem.

Cheng and Gordon (1994) again consider the batching concept on a single machine scheduling environment. The objective is to minimize the total distribution cost that mainly depends on the number of batch deliveries. The authors provide a dynamic programming approach which yields two pseudopolynomial algorithms when the number of batches has a fixed upper bound. In this study the authors also consider a special case where the processing times of the jobs in the same batch are assumed to be equal. A polynomial algorithm for this special case is presented.

2.2.2. Single Machine and Multiple Customer Areas

There are not many studies that fall into this category. The main reason for this is that routing decisions may be involved when there are multiple customers at different locations.. Also the properties of the batches are again determined according to these routing decisions. Although we consider identical plants at different locations and a single customer area in our study, this case has considerable importance for our research. In order to get an intuition for our study, it is possible to think the reverse of this multiple customer areas case.

Considering multiple customer areas is a relatively new concept and the study of Chang and Lee (2004) is among the first papers in this area. Two customer areas are considered and the finished products are delivered with a single and capacitated vehicle allowing milkrun. The distances from the machine to each customer are prespecified. The objective is to minimize the time by which all the jobs reach their customers. All jobs delivered in one shipment are called as a batch and it is stated that if the assignment of the jobs to the batches is known, the problem can easily be converted to a two-machine flow shop problem with the objective of minimizing the makespan where the vehicle is seen as the second machine. For this special case Johnson's rule solves the problem optimally. However, the original problem is shown to be NP-hard in the strong sense. A heuristic is provided that yields solutions with a worst case error bound of 7/4.

Pundoor and Chen (2005) consider the case with *m* customers positioned at different locations. It is assumed that only direct shipping from the supplier to the customer is used, i.e. no milkrun. Therefore, only the jobs associated to the same customer can be delivered together in a shipment. A combined objective function which considers both the maximum tardiness and the total distribution cost is used. It is shown that for arbitrary number of customers, the problem is NP-hard even for the special case where the processing times and due dates are agreeable (i.e. if $p_i \leq p_j$ then $d_i \leq d_j$). A fast heuristic which is capable to generate near optimal solutions is provided. One of the most important properties of this study is that they have showed that using the integrated production-distribution approach is more advantageous when compared to the approach of optimizing production and distribution sequentially with little or no integration.

The study of Chen and Vairaktarakis (2005) is important as it considers all the machine configuration and customer area cases. In this section, we will only review the single machine and multiple customer case. They allow milkrun in their study and determine the distances of the customers to the processing facility according to the milkrun routes. Their objective is to minimize total distribution cost and mean (or maximum) delivery time simultaneously and they provide efficient dynamic programming algorithms to solve these problems optimally.

Li et al. (2005) is one of the studies we base our research on. This study considers *m* customers at different locations and aims to minimize the total flow time. First, the problem is proven to be NP-hard in the strong sense. For the case where milkrun is allowed, first a pseudopolynomial dynamic programming algorithm for two-customers is provided and then generalized to multiple customers. Another dynamic programming algorithm for the case with only direct shipments is presented. The authors conclude that the computational complexity is lowered when the deliveries are restricted to direct shipments.

2.2.3. Multiple Machines and Single Customer Area

This is the second most widely studied case after single machine and single customer area. In this section, we consider mainly two types of problems. The first type is with a single production plant that has different machine configurations such as parallel machines, flow shops, and job shops. The second type considers multiple production plants dispersed at different geographic points where each plant can be viewed as a single machine. The problem studied in this thesis is among the second type. In our research we consider m identical machines at different locations and a single customer area. Delivery to the customer can be made from all of the plants via a single truck with a certain capacity. To the best of our knowledge, no one has studied this problem before. In our problem only direct shipments are allowed. Most of the studies in the relevant literature consider direct shipments. The studies that consider milkruns will be mentioned both in this and the following section.

Most of the work issuing multiple machines and a single customer area address the problems where the machines are located at the same location, i.e. parallel or flow-shop machines. The studies with parallel machines can be regarded as special cases of our problem where the distances between the machines are negligible and products of different machines can be transported on the same shipment.

Hall and Shmoys (1989) consider two multiple machine problems in their study. In the first one, their objective is to minimize maximum lateness of jobs with respect to release dates and delivery times in a parallel machine environment. They provide a polynomial approximation scheme for the case without precedence constraints and a 2-approximation algorithm for the case considering precedence. A two-machine flow shop environment is considered for the second problem. They have specified the special case where all release dates equal zero and stated that Johnson's rule solves the problem optimally. Then as in the parallel machine case, they provide a polynomial approximation scheme for the general problem. Woeginger (1994) showed that even for the case without release dates the problem

is NP-hard when the objective is to minimize the makespan. Two heuristics with the best one having a worst-case performance guarantee of $2-2/(m+1)$ are provided.

In the study of Wang and Cheng (2000), it is shown that parallel machines problem with the objective of minimizing total flow time and delivery cost is NP-hard in the strong sense and a dynamic programming algorithm is provided to solve it. Also polynomial time algorithms to solve the special cases where the job assignments to machines are given or the processing times are equal. Chen and Vairaktarakis (2005) has the objective of minimizing mean/maximum delivery time and total distribution cost and proves the problems to be NP-hard. The study also provides polynomial algorithms for the special cases where they consider only the total distribution cost.

As in the single machine and single customer case, Hall and Potts (2005) consider a constraint *T* that denotes the minimum time interval between two consecutive deliveries. It is shown that the recognition versions of the problems $P2|\ |Dy+\sum C_j$ and $P2|T|Dy+\sum C_j$ are binary NP-complete (*Dy* denotes the delivery cost). Pseudopolynomial algorithms are provided for the problems $P2|\ |Dy+L_{\max}$, $P2|T|Dy+L_{\max}$, $P2|\ |Dy+\sum w_jU_j$ and $P2|T|Dy+\sum w_jU_j$. Chang and Lee (2004) also study the two parallel machines problem but with a single and capacitated truck and with the objective of minimizing the makespan. A heuristic with a worst-case performance ratio bound of 2 is presented in this study.

Lee and Chen (2001) study a two-machine flow shop environment with delivering products to a single customer area with the objective of minimizing the makespan. This study is different in the way that it considers both the transportation inside a manufacturing facility (type-1 transportation) and the transportation between the facility and the customer (type-2 transportation). It is shown that type-1 transportation problems, where there exists a single truck that can carry only one job at a time or more than three jobs, are strongly NP-hard. A dynamic programming algorithm that solves the problem in polynomial time is provided for

the case where the processing times of all jobs are equal for the same machine. For type-2 transportation where jobs are first processed on a two-machine flow shop, then delivered to the customer by a single truck, the problem is shown to be strongly NP-hard when the truck can carry only one or more than four jobs at a time and they provide a heuristic for this problem.

Although most of the studies in the literature discussed in this section consider the problems with multiple machines at the same plant, there are also a few papers that address the problems with plants at different locations. These studies are very important for our research since they use the same machine and customer settings with our problem. However, since all of these studies consider infinitely many trucks, our research remains to be the first study in this machine and customer setting that takes the truck availability into account. Chen and Pundoor (2006) deserve particular attention since this is the first study to consider decentralized machines in this context. The authors define a problem where the products are produced on plants at different locations, and transported to a single warehouse assuming that there is a truck available at each plant at any time. Assigning a certain cost for each delivery trip, the aim of the study is to minimize four different objective functions in this problem setting:

- Problem 1: Minimizing a weighted sum of the total lead time and total cost
- Problem 2: Minimizing the total cost subject to the constraint that the total lead time is no longer than a given threshold
- Problem 3: Minimizing a weighted sum of the maximum lead time and total cost
- Problem 4: Minimizing the total cost subject to the constraint that the maximum lead time is no longer than a given threshold

All these four problems are shown to be NP-hard, and heuristics for those problems are developed in their study. Also, polynomial time exact algorithms are presented for some special cases. Li and Ou (2007) study a similar problem to the one in Chen and Pundoor

(2006); with the distinction of considering jobs with multiple tasks where each task of a job must be processed by a specific machine.

Both Chen and Pundoor (2006)'s and Li and Ou (2007)'s works are designed for time-sensitive products that have a large variety, a short life cycle, and are sold in a very short selling season. In this setting, using third-party carriers and hence assuming infinitely many trucks is logical. However, the work in our study is generally designed for companies using their own fleet or for small companies that might require outsourcing. Therefore, assuming infinitely many trucks would not be a logical assumption in our setting. On the other hand, having a single truck and taking truck availability into account is a more realistic assumption for our research setting.

2.2.4. Multiple Machines and Multiple Customer Areas

As the number of multi-facility companies increase, the need for the realistic models that consider multiple machines also increase. When it is assumed that shipments to the customers are made directly from the facilities without using a warehouse, it is reasonable to consider multiple customer areas. One of the studies that consider this case is the paper of Chen and Vairaktarakis (2005). They consider parallel machines to process the jobs and delivery of the products to the customers at different locations allowing milkrun. The problems under the objective of minimizing total distribution cost and mean/maximum delivery time are shown to be NP-hard and heuristics are provided for these problems.

The second and to our knowledge the last paper that considers this case is Li and Vairaktarakis (2007). The authors study an integrated production and distribution scheduling system where the jobs are processed on two flow shop machines which are located at the same facility and their output bundled together for delivery. The objective is to minimize the total delivery cost and the customers' waiting costs that depend on the arrival times of the

jobs at the customers. Both direct shipments and milkruns are considered. A polynomial time approximation scheme is provided for the special case with the objective of minimizing total flow time. For the original problem, heuristics with constant worst-case error bounds are presented.

Figures 2.1 and 2.2 show the classification for the literature presented in this chapter.

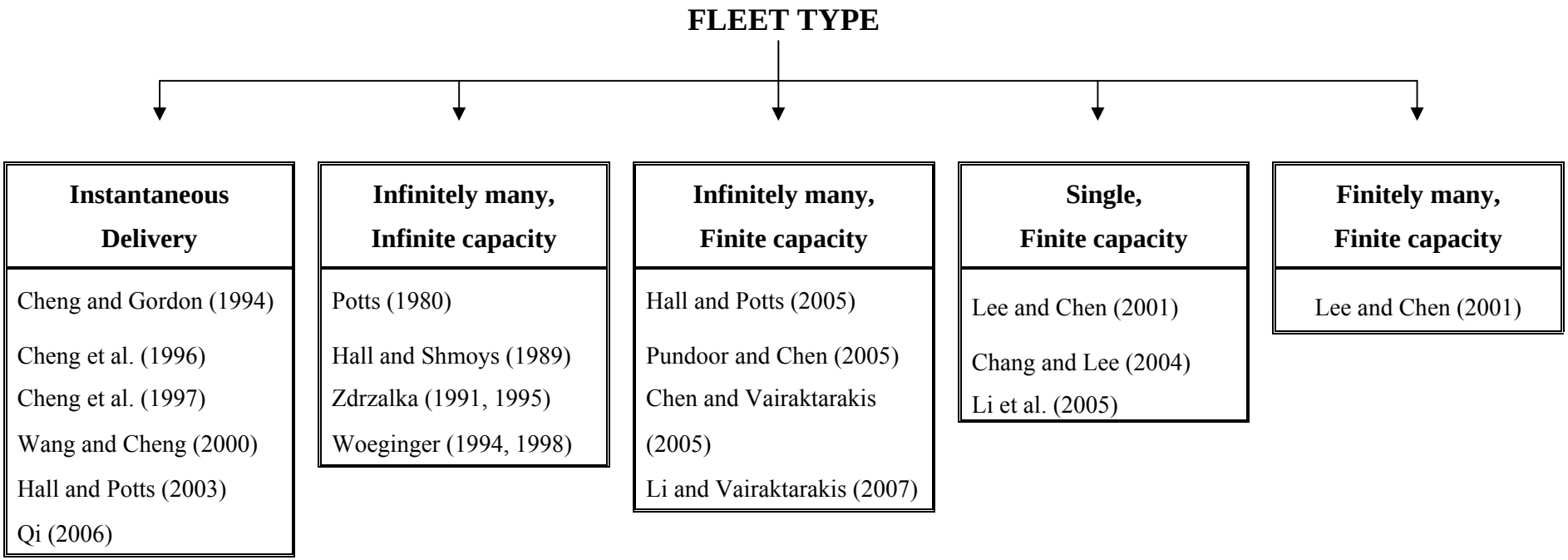

Figure 2.1: Literature classification for fleet type

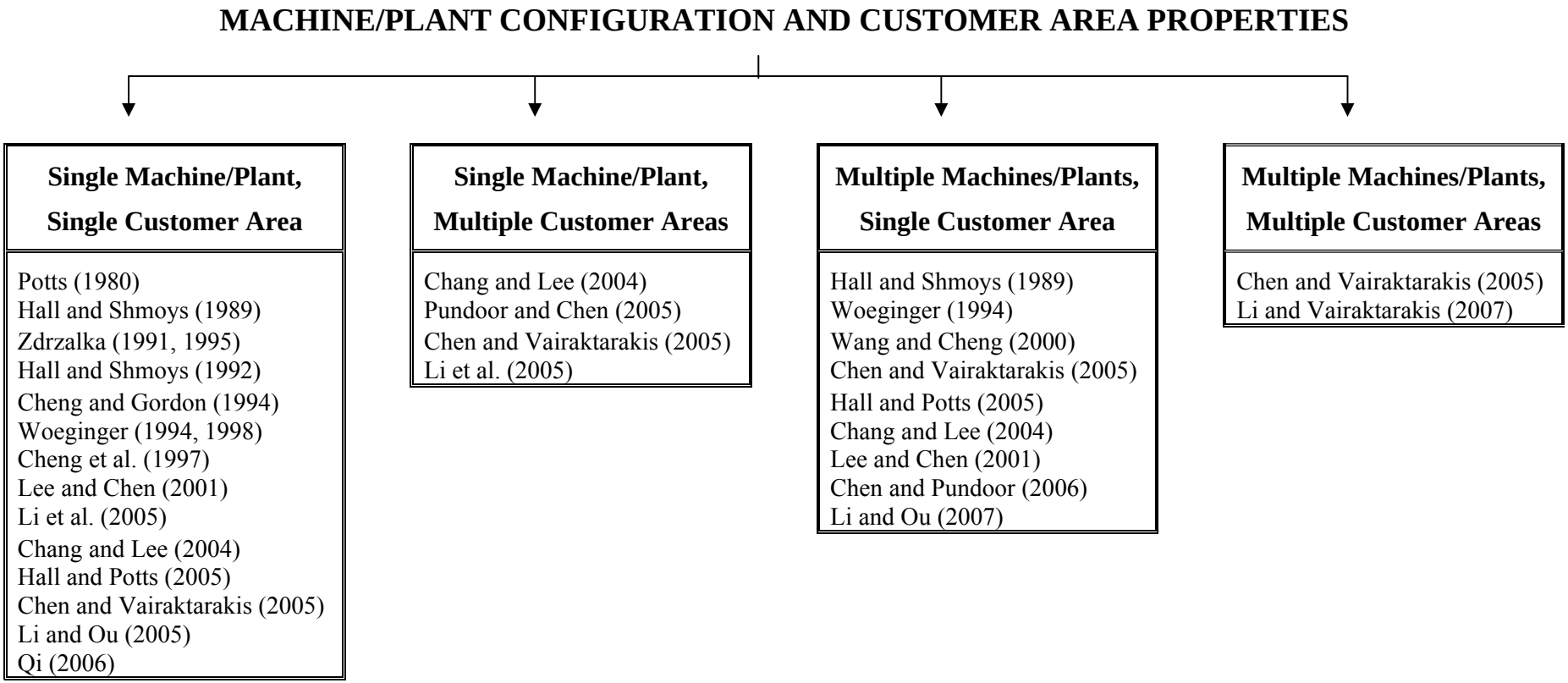

Figure 2.2: Literature classification for machine/plant configuration and customer area properties

Chapter 3

PROBLEM DESCRIPTION AND NOTATION

In this thesis, we work on a problem that considers simultaneous scheduling of production and transportation operations of a multi-plant manufacturer. The plants are located at different geographical points. The products produced in these plants are distributed to a single customer area using a single capacitated truck. Here the customer area may denote a single warehouse as well as multiple customers that are in close proximity to each other.

Consider a set of *n* jobs, $N = \{1, 2, ..., n\}$, each of which is to be processed in exactly one of the m plants, $M = \{1, 2, ..., m\}$, positioned at different locations. We assume that plants are identical (there is a dedicated production line in each plant for the customer area), and every plant can process all jobs. It takes p_j units of processing time to produce job j, $j \in N$, on any of the plants. Job j is denoted as J_j. Each job needs to be processed on only one of the plants without interruption. Preemption is not allowed. Each finished job should be transported from the plant on which it is produced to the customer. A job cannot be transported unless it has completed its processing in the plant. The distance of plant i, $i \in M$, to the customer is denoted as d_i and given in time units. Hence, in other words, it takes the transporter d_i units of time to go from plant i to the customer. It is assumed that the travel time from a plant to the customer is equal to the travel time from the customer to that plant. Distribution of finished jobs from the plants to the customer is done by a single

truck which can carry up to K jobs. Loading and unloading times are assumed to be negligible. Only direct shipments are allowed. To the best of our knowledge, in practice, companies with multiple plants try to allocate those plants so that they will be able to serve as many warehouses as possible. Since they do not require the plants to be in close proximity of each other, it can be assumed that those plants may be distant to each other. Therefore, our assumption of direct shipments is reasonable. Initially, the truck is assumed to be located at the customer area and for convenience each shipment (i.e. each visit of the truck to a plant) will be called a *trip*. A simple representation of plant locations and their distances to the customer area can be seen in Figure 3.1 for an example with 4 plants.

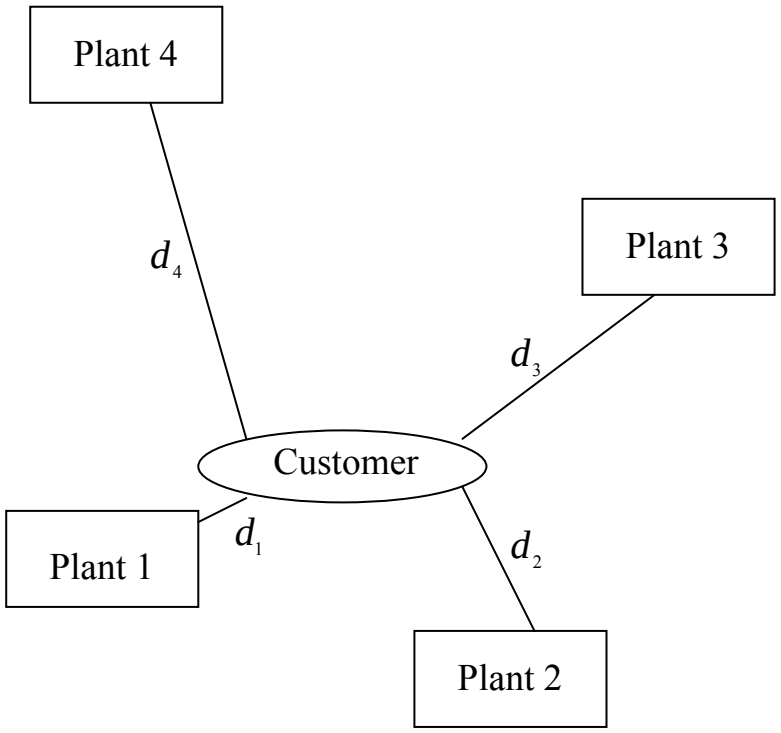

Figure 3.1: A representation of plant locations and distances to the customer area

Without loss of generality, throughout this thesis, the plants are named according to their distances to the customer area. The plant with the smallest distance is called Plant 1, the one with second smallest distance is called Plant 2, and so on. Again without loss of generality, jobs are named according to their processing times. The job with the smallest

processing time is referred to as job 1, the one with second smallest processing time is called job 2, and so on. Hence, $d_1 \leq d_2 \leq .... \leq d_m$ and $p_1 \leq p_2 \leq .... \leq p_n$.

For job j, produced on plant i, to be transported from that plant to the customer at time t, there are two conditions to be satisfied:

- Job j needs to have completed its processing on or before time t,
- The truck should arrive at plant i on or before time t and should still be there at time t.

As common in the relevant literature on integrated scheduling of production and distribution operations, the completion time of job j is determined as the time that job reaches the customer. In our study, the objective is to minimize the total completion time of all jobs. A solution constitutes the following:

- Assignment of the jobs to the plants
- Scheduling of production of jobs at each plant
- The route and schedule of the truck
- Assignment of the jobs to the *trips*

Suppose that jobs are assigned to a plant set M', $M' \subseteq M$. The route of the truck is the order truck visits the plants in M'. For example, if the truck visits first Plant 2, then Plant 1, and then Plant 2 again, the route of the truck is Plant 2 – Plant 1 – Plant 2. Determining the schedule of the truck means determining the time truck leaves each plant on its route.

In this study, we consider a single planning horizon and we aim to send the jobs to the customer as soon as possible within this planning horizon. Viewing the customer area as a warehouse, the length of this planning horizon depends on the demand amount of the warehouse.

In Chen (2008), MP notation is used for multiple decentralized plants. Denoting the completion time of job $j \in N$ as C_j, and using the three-field notation as in Lee and Chen (2001), our problem can be denoted as $MP \to 1|v = 1, K \geq 1|\sum C_j$. Here $MP \to 1$ means from multiple plants to a single customer area, and $v = 1, K \geq 1$ means using a single truck with a capacity of K. The following theorem holds for our problem.

Theorem 1. *There exists an optimal schedule that satisfies the following conditions for each plant:*

(i) *If J_i is processed earlier than J_k, then J_i leaves the plant no later than J_k.*

(ii) *Jobs are processed in non-decreasing order of processing times in each plant.*

(iii) *There exists no idle time between job processing in any given plant.*

(iv) *The transporter either leaves the plant immediately as it arrives or at the completion time of a job.*

Proof:

(i) Suppose that there exists an optimal schedule in which J_i is processed earlier than J_k but leaves the plant after J_k. While having the same transportation schedule, if we interchange the processing positions of these jobs at the plant such that J_k is processed earlier than J_i, the total completion time will not change. If we apply the same interchange procedure to all necessary job pairs, we will have a transportation sequence which is the same with the processing sequence.

(ii) Consider an optimal schedule where jobs are not processed in non-decreasing order of processing times (SPT order). Then there exists at least two adjacent jobs J_i and J_k where J_i is processed before J_k and $p_i > p_k$. If J_i and J_k are transported on the same trip, then interchanging the processing positions of these jobs at the plant will not increase total completion time. However, if J_i and J_k are not transported on the same trip, interchanging

the positions of these jobs may decrease the total completion time by decreasing the start times of the trips on which these jobs are transported.

(iii) If there exists idle time between job processing, we can move the jobs succeeding the idle period earlier without increasing the total completion time.

(iv) Suppose the vehicle leaves the plant neither as soon as it arrives there nor at the completion time of processing of a job. Then we can move subsequent trips earlier to either the arrival time of the truck at the plant or to the process completion time of the most recently loaded job and in this way decrease the total completion time.■

The following section aims to more precisely convey the parameters given in a problem instance as well as the structure of a desirable solution by considering a numerical instance.

3.1. Sample Problem

Suppose that we have 10 jobs to be processed at a subset of 3 plants, and to be transported with a truck that carries at most 4 jobs in a trip. First, the plants are sorted in non-increasing order of their distances to the customer and reindexed, and the same procedure is applied for the jobs with respect to their processing times. After sorting, the distances of the plants can be listed as $d = \{d_1, d_2, d_3\} = \{26, 30, 37\}$ whereas the processing times of the jobs can be listed as $p = \{p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9, p_{10}\} = \{6, 6, 10, 11, 22, 25, 27, 29, 33, 34\}$. As can be seen in Figure 3.2, the optimal solution to this problem assigns jobs $\{J_1, J_2, J_3, J_4, J_6, J_9, J_{10}\}$ to Plant 1, and the rest to Plant 2. As have been stated in Theorem 1, the jobs are processed according to SPT rule at each plant.

Trip 1: At time 0, the truck leaves the customer and goes to Plant 1 and arrives at that plant at time $d_1 = 26$. Jobs $\{J_1, J_2, J_3, J_4\}$ are transported on the first trip and since $p_1 + p_2 + p_3 + p_4 = 33 > 26 = d_1$, truck waits for the completion of the processing of J_4, and leaves the plant at time 33. Truck arrives at the customer at time $33 + 26 = 59$. Hence the completion time for jobs $\{J_1, J_2, J_3, J_4\}$ is 59.

Trip 2: On the second trip truck goes to *Plant2*, and arrives there at time $59 + 30 = 89$. It takes jobs $\{J_5, J_7, J_8\}$ and since $p_5 + p_7 + p_8 = 78 < 89$, the truck will leave Plant 2 as soon as it arrives and will turn back to the customer at time $89 + 30 = 119$. This means that the completion time for jobs $\{J_5, J_7, J_8\}$ is 119.

Trip 3: Truck makes the third trip to Plant 1 again and it arrives at Plant 1 at time $119 + 26 = 145$. Since $p_1 + p_2 + p_3 + p_4 + p_6 + p_9 + p_{10} = 125 < 145$, truck leaves Plant 1 as soon as it arrives there and transports jobs $\{J_6, J_9, J_{10}\}$. The truck turns back to the customer at time $145 + 26 = 171$ and hence the completion time for jobs $\{J_6, J_9, J_{10}\}$ is 171.

Instead of summing individual completion times of the jobs, it is possible to calculate the total completion time by summing the contributions of each trip, where the contribution of a trip is calculated by the multiplication of the number of jobs transported on that trip and the arrival time of the truck to the customer at the end of that trip. Therefore, total completion time can also be shown as $\sum C_j = 4 \cdot 56 + 3 \cdot 119 + 3 \cdot 171 = 1106$. Here the coefficients $\{4,3,3\}$ are the number of jobs transported on and the multipliers $\{56,119,171\}$ are the completion times of trip 1, trip 2 and trip 3, respectively.

It should be noted that, in the optimal solution of this sample problem, Plant 3 is not used and all the jobs are produced on Plant 1 and Plant 2. The reason for this situation is that Plant 3 is further than other plants and since it is possible to produce all the jobs in closer plants, it is not logical to use Plant 3. However, this may not be the case for each instance.

Especially for the instances which have larger processing times when compared to this problem, it may be better to produce also in Plant 3.

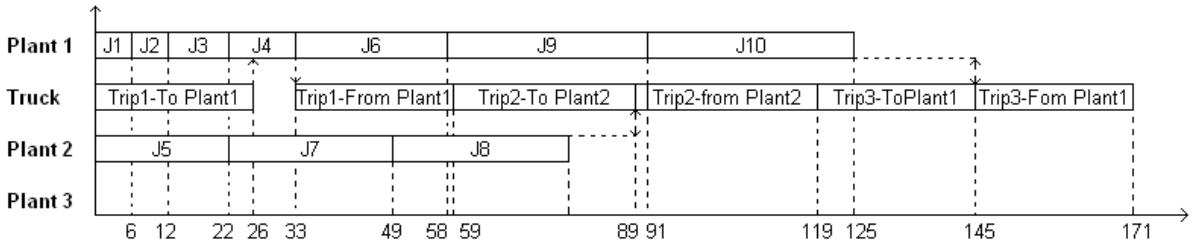

Figure 3.2: Gantt chart for the optimal solution of sample problem with 10 jobs and 3 plants

Chapter 4

METHODOLOGY

In Chapter 3, the problem studied in this thesis was defined. In this chapter, we present the constructive heuristics and exact methods to solve this problem. To facilitate the assessment of heuristics, we develop fast lower bounds. In order to be able to provide a good understanding of the methods proposed we will follow a path from simpler cases referred to as special cases, to the main problem. The first special case we investigate will be the problem where we assume the assignments of the jobs to the plants are given. A dynamic programming algorithm will be presented for this problem. Then we will study the problem where we have the constraint of sending the truck fully loaded in its all trips, which we prefer to name as *Fully Loaded Trips Problem*. A special case of this problem, where it is known that last trip will be partially loaded, will be explained and the methods to generate exact solutions, heuristic methods to obtain near optimal solutions for both problems will be presented. Since these problems are NP-hard, for which proofs will be provided, it is hard to obtain exact solutions in a reasonable amount of time. Therefore, we have also generated lower bounds to measure the efficiency of our heuristic methods. The main problem, referred to as *General Problem*, occurs when we do not have the fully loaded trucks constraint. Exact solution methods, heuristic algorithms and lower bounds will be presented for both this main problem and its special case, where the truck leaves the plants it visits on each trip as soon as it arrives there.

4.1. Job Assignments Given Problem

In this problem, we assume that the assignments of the jobs to the plants are given so that we know which job will be processed on which plant. Therefore, the only question we need to answer is which route the truck should follow and which jobs it should take from the plant it visits. The truck leaves the plant either as soon as it arrives there or at the completion time of the last job transported on that trip. Hence, if which jobs to be transported on a trip is determined, then it can be determined whether the truck leaves the plant immediately or not. This case is mostly applicable when there are eligibility conditions so that each job can be produced only on a certain plant set.

In some cases, the distances of the plants to the customer may be equal or close to each other. Then for such problems, it is possible to assume that the distances of each plant to the customer are equal. In such a case, the following dynamic programming algorithm may be applied.

4.1.1. Plants at Equal Distances

First, we need to give some relevant notation:

- n_i : number of jobs assigned to plant i, $i = 1,...,m$ such that $\sum_{i=1}^{m} n_i = n$
- C_j^i : the completion time of the j^{th} job at plant i
- d : the distance from a plant to the customer

The truck will leave the customer at time 0 and will go to a plant. Since all plants have the same d distance to the customer, the truck will be at some plant at time d. As stated in Theorem 1(iv), it will leave the plant either immediately or will wait until the completion of all jobs transported on that trip. Then it will return to the customer in d units of time. It will leave the customer again as soon as it arrives to the customer and start a new

trip. Therefore, there can only be a finite number of time points the truck can depart from the customer, i.e. the time points a trip can start. Let T be the set of possible departure times of the truck from the customer. Then each departure time t, $t \in T$, can be obtained using the following formula:

$$(C^1_{j_1} + a_1 \cdot d) + (C^2_{j_2} + a_2 \cdot d) + ... + (C^m_{j_m} + a_m \cdot d) + q \cdot 2 \cdot d$$

where $C^i_0 = 0$ for $i = 1,...,m$; $j_i = 0,1,....,n_i$ for $i = 1,...,m$; $a_i = 0,1$ for $i = 1,...,m$ and $q_i = 0,1,2,....,n$ for $i = 1,...,m$.

Let $f(i_1, i_2, ..., i_m, t)$ denote the minimum possible total completion time of the jobs $\{J_{i_1}, J_{i_1+1}, ..., J_{n_1}\}, \{J_{i_2}, J_{i_2+1}, ..., J_{n_2}\}, ..., \{J_{i_m}, J_{i_m+1}, ..., J_{n_m}\}$ given that the truck departs the customer at time t to complete the transportation of those jobs, where $t \in T$, $i_1 = 1,...,n_1$, $i_2 = 1,...,n_2$, …, $i_m = 1,...,n_m$. In other words, since some of these jobs have been already transported to the costumer in the previous trips, the transportation of the jobs $\{J_{i_1}, J_{i_1+1}, ..., J_{n_1}\}, \{J_{i_2}, J_{i_2+1}, ..., J_{n_2}\}, ..., \{J_{i_m}, J_{i_m+1}, ..., J_{n_m}\}$ will be completed once the trip with starting time t is over.

If we denote h_i to be the number of jobs transported from plant i to the customer on the trip starting at time t, we can write the recursive function as follows:

$$f(i_1, i_2, ..., i_m, t) = \min \left\{ \begin{array}{c} \min\limits_{0 \le h_1 \le \min(n_1 - i_1 + 1, K)} \{h_1 \cdot A_1 + f(i_1 + h_1, i_2, ..., i_m, A_1)\}, \\ \min\limits_{0 \le h_2 \le \min(n_2 - i_2 + 1, K)} \{h_2 \cdot A_2 + f(i_1, i_2 + h_2, ..., i_m, A_2)\}, \\ . \\ . \\ \min\limits_{0 \le h_m \le \min(n_m - i_m + 1, K)} \{h_m \cdot A_m + f(i_1 + h_1, i_2, ..., i_m, A_m)\} \end{array} \right\}$$

where $A_k = \left\lceil \max(t+d, C^k_{i_k+h_k-1}) + d \right\rceil$ for $k = 1,....,m$. The boundary condition is $f(n_1+1, n_2+1,...., n_m+1, t) = 0$ for all $t \in T$, such that $t \geq \sum_{j=1}^{n} p_j + m \cdot d$, and the objective is $f(1,1,....,1,0)$.

Lemma 4.1: The overall complexity of the dynamic programming for the problem with known job assignments and plants at equal distances to the customer is $O(2^m n^{2m+2})$.

Proof: Using the formula $(C^1_{j_1} + a_1 \cdot d) + (C^2_{j_2} + a_2 \cdot d) + ... + (C^m_{j_m} + a_m \cdot d) + q \cdot 2 \cdot d$ we can obtain each departure time. Each j_i can have $(n_i + 1)$, each a_i can have two and q can have $(n_i + 1)$ different values. Hence there are $(n_1+1) \cdot 2 \cdot (n_2+1) \cdot 2 \cdot ..... \cdot (n_m+1) \cdot 2 \cdot (n+1)$ possible departure times. Therefore, $|T| = O(2^m n^{m+1})$. In the recursive formulation, the number of possible $(i_1, i_2, ..., i_m, t)$ combinations is $|T| \cdot \prod_{i=1}^{m} (n_i + 1)$, meaning there are $O(n^m |T|) = O(2^m n^{2m+1})$ states to evaluate. When m is fixed, it takes $O(n)$ time to evaluate each $f(i_1, i_2, ..., i_m, t)$ and hence the overall complexity is $O(2^m n^{2m+2})$. Since in real life m does not grow exponentially and usually is a small number, this algorithm works in polynomial time when m is fixed.

4.1.2. Plants at Different Distances

In this case, we will assume that plants do not have the same distances to the customer. Let d_i be the distance from plant i to the customer where $i = 1,..., m$. The set of the possible departure times of the truck from the customer can be found using the following formula:

$$(C^1_{j_1} + a_1 \cdot d_1) + (C^2_{j_2} + a_2 \cdot d_2) + ... + (C^m_{j_m} + a_m \cdot d_m) + q_1 \cdot 2 \cdot d_1 + q_2 \cdot 2 \cdot d_2 + ... + q_m \cdot 2 \cdot d_m$$

where a_i, j_i, $C^i_{j_i}$ are as described in the previous case and $q_i = 0,1,2,....,n_i$ for $i = 1,...,m$.

The recursive function generated for the problem in the previous section, i.e. the problem with equal distance plants, can be used for this problem with a slight modification of A_k values such that $A_k = \left\lceil \max(t + d_k, C^k_{i_k + h_k - 1}) + d_k \right\rceil$ for $k = 1,....,m$. The boundary condition and the objective are the same as in the equal distance problem in Section 4.1.1.

Lemma 4.2: The overall complexity of the dynamic programming algorithm for the problem with known job assignments and plants at different distances to the customer is $O(2^m n^{3m+1})$.

Proof: There are $(n_1 + 1) \cdot 2 \cdot (n_2 + 1) \cdot 2 \cdot ..... \cdot (n_m + 1) \cdot 2 \cdot (n_1 + 1) \cdot (n_2 + 1) \cdot ..... \cdot (n_m + 1)$ possible departure times. Hence, $|T| = O(2^m n^{2m})$. In the recursive function, the number of possible $(i_1, i_2, ..., i_m, t)$ combinations is $|T| \cdot \prod_{i=1}^{m} (n_i + 1)$ and therefore there are $O(n^m |T|) = O(2^m n^{3m})$ states to evaluate. It takes $O(n)$ time to evaluate each state. Therefore, overall complexity is $O(2^m n^{3m+1})$. When m is fixed, this problem can be solved in polynomial time.

4.2. Fully Loaded Trips Problem

One of our motivations in studying the problem of $MP \to 1|v=1, K \geq 1|\sum C_j$ is that it is a problem faced by the soft drink manufacturer mentioned in Chapter 1. This manufacturer is producing physically sensitive products and wants to transport them with less damage. For example, glass bottles may be broken when scattered around in the truck and the cans may be damaged when shaken. The easiest solution preferred by this soft drink manufacturer in the transportation of these products is sending the trucks full so that the damage to the products will be minimized. Also, companies feel they utilize truck capacity better when they send the trucks fully loaded. Hence, we found it beneficial to study the special case of our problem where we have the constraint of sending the truck fully loaded, i.e. loading K jobs, in all trips. However, since the number of jobs may not be divisible by the capacity of the truck, we may end up with a partially loaded trip. Here, the truck being partially loaded means that it would transport fewer than K jobs on that trip. In the main problem, we are required to answer how many jobs will be transported on a trip and which jobs will constitute the jobs being transported. In fully loaded trips problem, instead of answering how many jobs to be transported on each trip, we need to determine which trip will be partially loaded.

Last Trip Partial Problem, is a special case of fully loaded trips problem. In this problem, as the name implies, the only partially loaded trip will be the last one. Clearly, a partially loaded trip will be required only if the number of jobs, n, is not divisible by the truck capacity K.

Before continuing with the solution methods for the problem having the last trip partial, we will first prove that the problem with fully loaded trips, except one if a partially loaded trip is required, is NP-hard in the ordinary sense in Theorem 2. In Corollary 2 it will be shown that Last Trip Partial Problem is also NP-hard in ordinary sense.

4.2.1. NP-Hardness Proofs for the Fully Loaded Trips Problem and Its Special Case Last Trip Partial Problem

In this section, to provide clarity, we denote the problem with fully loaded trips, and one partial trip if needed, as $MP \to 1|v=1, K \geq 1, FullTrips|\sum C_j$. Its special case where the last trip is partial, if a partially loaded trip is required, is denoted as $MP \to 1|v=1, K \geq 1, LastPartial|\sum C_j$. To prove the NP-hardness of the problems in this chapter, we use well known PARTITION problem defined as follows:

Given a finite set A and a size $s(a) \in \mathbb{Z}^+$ for each $a \in A$, is there a subset $A' \subseteq A$ such that $\sum_{a \in A'} s(a) = \sum_{a \in A - A'} s(a)$?

Theorem 2: Problem $2P \to 1|v=1, K \geq 1, FullTrips|\sum C_j$ is NP-hard.

Proof: Given an instance of the PARTITION problem, $A = \{a_1, a_2, ..., a_{2h}\}$, we construct the following instance of our problem.

Number of jobs $n = 4h$, and jobs set $N = H \cup \{2h+1\} \cup \{2h+2, 2h+3, ..., 4h\}$ where $H = \{1, 2, ..., 2h\}$

Number of plants $m = 2$

Truck capacity $K = 2h$

Processing times $p_j = s(a_j)$ for $j \in H$, $p_{2h+1} = \sum_{j \in H} p_j = 2B$,

$$p_j = 0 \text{ for } j = \{2h+2, 2h+3, ..., 4h\}$$

Plant distances $d_1 = d_2 = d = \frac{\sum_{j \in H} p_j}{2} = B$

Threshold on total completion time $C = 12Bh$

$\rightarrow$ If there is a solution to the PARTITION instance, we show that there is a schedule for the above constructed instance with a total completion time of no more than C. Let G be a subset of H that solves the PARTITION instance. Since $|H| = 2h$, $|G| < K$. Produce $K - |G|$ of the jobs with zero processing time and all jobs in G on Plant 1. Schedule these jobs in non-decreasing order of processing times (SPT order) on Plant 1, as stated in Theorem 1(ii). Assign the rest of the jobs to Plant 2 and schedule them in SPT order. Hence, the K jobs assigned to Plant 1 will complete their processing at time $B = \frac{\sum_{j \in H} p_j}{2}$ and the other K jobs, assigned to Plant 2, will complete their processing at time $3B = \sum_{j \in H} p_j + \frac{\sum_{j \in H} p_j}{2}$. Truck will leave the customer at time 0 and will arrive at Plant 1 at time d. Since $d = B$, it will take all K jobs assigned to that plant and will return to the customer at time $2d$. Then truck will leave the customer and arrive to Plant 2 at time $3d = 3B$. Again since all K jobs have already been processed, truck will take them and arrive at the customer at time $4d$. Hence, total completion time will be $2Kd + 4Kd = 6Kd = 12Bh$.

$\leftarrow$ If there exists a schedule with total completion time of no more than C, then there must be a solution to PARTITION instance. Since there are $2K$ jobs, it is for sure that there will be two trips. Consider two cases:

Case 1: Both trips visit the same plant. In this case all of the jobs will be processed at one of the plants in SPT order. Truck arrives at the plant at time d and up to that time all of $K - 1$ jobs with zero processing time will have been processed. Then depending on,

$p = \min_{j \in H} p_j$, truck will leave plant at time $d' \geq d$. It will arrive at the customer at time $d' + d$ and will go back to the plant at time $d' + 2d$. Since the completion time of all jobs is $2\sum_{j \in H} p_j = 4B = 4d$ and $d' < 2d$, truck will leave plant at time $4d$ and will arrive at the customer at time $5d$. Hence, total completion time is going to be $K(d' + d) + 5Kd \geq 7BK > C$. Since total completion time of this case is greater than C, the truck should not visit the same plant in both trips.

Case 2: One of the trips is to Plant 1 while the other is to Plant 2. Since Case 1 is shown to yield a total completion time greater than C, this case is the only possible case. Assuming without loss of generality that the first trip is to Plant 1, one of the questions is on which plant J_{2h+1}, i.e. the job with processing time of $2d$, will be produced. If J_{2h+1} is produced at Plant 1, at best, the jobs with zero processing time will be produced there and the truck will leave Plant 1 at time $2d$, arriving at the customer at time $3d$. On its second trip, truck will arrive at Plant 2 at time $4d$ and will return to the customer at time $5d$. Hence, total completion time will be at least $3Kd + 5Kd = 8BK > C$. Therefore, J_{2h+1} will be assigned to the plant to which second trip is made.

Assuming without loss of generality that the first trip is to Plant 1, let M denote the set of jobs that are assigned to Plant 1 such that $M \subseteq H$. Even if the truck leaves the plant as soon as it arrives there on each trip, total completion time will be C. Since we want to obtain a schedule with total completion time of no more than C, it is required that K jobs are produced on each plant on or before the time truck arrives at that plant. Hence, we require $\sum_{j \in M} p_j \leq d = B$ and $\sum_{j \in H \setminus M} p_j + p_{2h+1} \leq 3d$. The latter can be re-expressed as $\sum_{j \in H \setminus M} p_j \leq d = B$. Since $\sum_{j \in H} p_j = 2B$, we have $\sum_{j \in M} p_j = \sum_{j \in H \setminus M} p_j = B$. Therefore, we have a solution to the PARTITION instance if there exists a schedule with a total completion time of no more than C.■

The following corollary immediately follows.

Corollary 1: $MP \to 1|v=1, K \geq 1, FullTrips|\sum C_j$ is NP-hard.

Corollary 2: Problem $2P \to 1|v=1, K \geq 1, LastPartial|\sum C_j$ is NP-hard.

Proof: Since Theorem 2 does not involve a partial trip the proof is valid also for this special case.

The problem $MP \to 1|v=1, K \geq 1, LastPartial|\sum C_j$ is also NP-hard since the problem $2P \to 1|v=1, K \geq 1, LastPartial|\sum C_j$ is shown to be NP-hard.

4.2.2. Last Trip Partial Problem

In this section we consider a problem that involves the constraint that the only partially loaded trip, when required, can be the last one. The need to decide which trip will be partially loaded increases the complexity of the problem. Thus, this constraint makes the problem easier. Managers in real life may also find it intuitive to send more jobs on the earlier trips and fewer jobs on the last trip when they wish to minimize the total completion time. With this motivation, an exact method to solve this problem will be given in the next section. Since the problem is NP-hard, it will be difficult to obtain optimum solutions in a reasonable amount of time and hence fast heuristic algorithms will also be proposed. Finally, a lower bound will be derived to facilitate the quality assessment of the heuristic algorithms.

4.2.2.1. Exact Solution Method for Last Trip Partial Problem

Parameters:

n : number of jobs

p_j : processing time of job j, for $j = 1, 2, ..., n$

d_i : distance between plant i and the customer, for $i = 1, 2, ..., m$

K : truck capacity

q' : number of trips required to transport all jobs, $q' = \left\lceil \frac{n}{K} \right\rceil$

q : number of fully loaded trips required, $q = \left\lfloor \frac{n}{K} \right\rfloor$

Note that if a partially loaded trip is not required, it will be that $q' = q$, otherwise $q' = q + 1$.

Assumptions:

We assume without loss of generality that jobs are indexed in non-decreasing order of their processing times. That is, $p_1 \le p_2 \le .... \le p_n$.

Decision Variables:

$$X_{ij} = \begin{cases} 1, & \text{if job } j \text{ is assigned to plant } i \\ 0, & \text{otherwise} \end{cases} \quad \text{for } i = 1, 2, ..., m \text{ and } j = 1, 2, ..., n$$

$$Y_{jk} = \begin{cases} 1, & \text{if job } j \text{ is transported on trip } k \\ 0, & \text{otherwise} \end{cases} \quad \text{for } j = 1, 2, ..., n \text{ and } k = 1, 2, ..., q'$$

$$W_{ik} = \begin{cases} 1, & \text{if plant } i \text{ is visited on trip } k \\ 0, & \text{otherwise} \end{cases} \qquad \text{for} \quad i = 1,2,....,m \ \text{ and } \ k = 1,2,....,q'$$

C_j = process completion time of job j at the plant it is assigned to, $j = 1,2,....,n$

Q_k = time truck reaches the customer at the end of the k^{th} trip, $k = 1,2,....,q'$

Mixed Integer Programming Model (MIP-1)

(1) minimize $K \cdot \sum_{k=1}^{q} Q_k + (n - q \cdot K) \cdot Q_{q'}$

subject to

(2) $\sum_{i=1}^{m} X_{ij} = 1, \qquad j = 1,2,..,n$

(3) $\sum_{k=1}^{q'} Y_{jk} = 1, \qquad j = 1,2,..,n$

(4) $\sum_{j=1}^{n} Y_{jk} = K\ , \qquad k = 1,..,q$

(5) $\sum_{j=1}^{n} Y_{j(q+1)} = n - q \cdot K$

(6) $W_{ik} \geq X_{ij} + Y_{jk} - 1, \qquad i = 1,..,m\ , \quad j = 1,..,n\ , \quad k = 1,..,q'$

(7) $\sum_{i=1}^{m} W_{ik} = 1, \qquad k = 1,..,q'$

(8) $$\sum_{l=1}^{j}(X_{il}\cdot p_l)\le C_j+(1-X_{ij})\cdot M_1,\qquad i=1,..,m\ ,\quad j=1,..,n$$

(9) $$Q_1\ge 2\cdot\sum_{i=1}^{m}(W_{i1}\cdot d_i)$$

(10) $$Q_k\ge Q_{k-1}+2\cdot\sum_{i=1}^{m}(W_{ik}\cdot d_i),\qquad k=2,...,q'$$

(11) $$(M_1+d_i)\cdot(2-X_{ij}-Y_{jk})+Q_k\ge C_j+d_i,\qquad i=1,...,m\ ,\quad j=1,...,n\ ,\quad k=1,...,q'$$

(12) C_j *is nonnegative real number*, $\qquad j=1,...,n$

(13) Q_k *is nonnegative real number*, $\qquad k=1,...,q'$

(14) $$X_{ij}\in\{0,1\},\qquad i=1,...,m\ ,\quad j=1,...,n$$

(15) $$Y_{jk}\in\{0,1\},\qquad j=1,...,n\ ,\quad k=1,...,q'$$

(16) $$W_{ik}\in\{0,1\},\qquad i=1,...,m\ ,\quad k=1,...,q'$$

Since only the last truck will be partially loaded, the critical information in determining the objective value is the times the truck arrives at the customer after each trip. The objective function shown in (1) formulates the total completion time as a function of these truck arrival times. Constraint set (2) ensures that each job is assigned to exactly one plant. Similarly, constraint set (3) ensures the assignment of each job to exactly one trip. Constraint set (4) enforces the condition that K jobs be transported on the first q trips and constraint (5) forces $(n-qK)$ jobs to be assigned on the last trip. In constraint (5) if q' was used instead of $(q+1)$, when no partially loaded trip is required, this constraint would try to assign 0 jobs to the last trip. Hence our model would not work since all jobs would not be transported. Constraint set (6) implies that if job j is transported on trip k, then the plant at which job j is produced should be visited on trip k. Constraint set (7) ensures that on each trip only one plant is visited therefore only direct shipments from each plant to the customer are permissible. Constraint set (8) requires a thorough explanation. According to Theorem

1(ii), all jobs assigned to each given plant are processed in SPT order. Since jobs are indexed in SPT order, the only jobs that affect the completion time of job j are the jobs with smaller indices at the same plant. Therefore, the completion time of job j is the summation of the processing times of these jobs and that of job j. M_1 in constraint set (8) is a large integer and its value is determined to be the sum of processing times of all jobs, i.e. $M_1 = \sum_{j=1}^{n} p_j$. Constraint (9) states that the truck cannot leave the plant it visits on its first trip before arriving there. Constraint set (10) ensures the same condition as in constraint (9) but for the trips other than the first one. In constraint set (11), instead of the term $\left(2 - X_{ij} - Y_{jk}\right)$, we can use $\left(1 - W_{ik}\right)$. This constraint set ensures that on any trip, if job j is transported on that trip, the truck cannot leave the plant job j is processed at before the completion of that job. Here, if a job is not transported on the k^{th} trip, the constraint becomes redundant. However, for each job transported on the k^{th} trip, this constraint must be satisfied.

4.2.2.2. Heuristic Methods for Last Trip Partial Problem

Preliminary computations indicate that the MIP model developed for the problem with last trip partially loaded is not able to provide results in a reasonable amount of time. Thus, we develop two fast heuristic methods to obtain near optimal solutions for this problem. Although the name of this problem is Last Partial Case, our heuristics are designed to handle situations where the number of jobs is divisible by the truck capacity, i.e. no partially loaded trip is required. Therefore, whenever we use the term $(n - qK)$ in order to represent the number of jobs in the partially loaded trip, it should be kept in mind that its value may be zero when a partially loaded trip is not required.

Heuristic 1 (LPH1):

There are two main principles supporting the mechanism of this procedure. We believe that the algorithm will present better intuitive appeal after a description of these two principles.

Principle 1 (*Batch the Jobs*): One of the main ideas underlying Heuristic 1 is that on each plant assigned jobs will be processed in SPT order. Therefore, all jobs are sorted in SPT order and then grouped into full batches of K jobs, and the partial batch with $(n-qK)$ jobs. A grouping example with a partially loaded trip is as shown below.

$$\underbrace{J_1, J_2, \ldots, J_K}_{Batch1}, \underbrace{J_{K+1}, J_{K+2}, \ldots, J_{2K}}_{Batch2}, \ldots\ldots\ldots, \underbrace{J_{qK+1}, J_{qK+2}, \ldots, J_n}_{Batch(q+1)}$$

In Heuristic 1, first Batch 1 will be assigned to a plant and that plant will be visited on the first trip. Then Batch 2 will be assigned to a plant and transported on the second trip and this will continue until all the batches are assigned to a plant and transported.

For ease of notation, as mentioned in Chapter 3, p_l will be used to denote the processing time of J_l.

Principle 2 (*Choose Potential Plants*): Initially none of the batches is assigned to any plant. We need to choose a plant to assign Batch 1. The processing of the jobs in Batch 1 takes $\sum_{j=1}^{K} p_j$ time. If Batch 1 is assigned to Plant 1, the first trip will end at time $\max\left(\sum_{j=1}^{K} p_j, d_1\right) + d_1$. If it is assigned to a farther plant i, the first trip will end at time $\max\left(\sum_{j=1}^{K} p_j, d_i\right) + d_i$. Since $d_i \geq d_1$, $\max\left(\sum_{j=1}^{K} p_j, d_i\right) + d_i \geq \max\left(\sum_{j=1}^{K} p_j, d_1\right) + d_1$. Therefore, to

minimize the completion time of the first trip, the truck should visit Plant 1 on the first trip. For the second trip, we know that Batch 1 is assigned to Plant 1 and no batches have been assigned yet to farther plants. With a similar calculation to the one made for the first trip, it is obvious that among the plants in the set {Plant 2, Plant 3, …, Plant m}, it is better to assign Batch 2 to Plant 2. Therefore, instead of thinking every plant as an alternative, only Plant 1 and Plant 2 are considered as alternatives for assigning Batch 2. To choose one of them, the completion time of the second trip is calculated for both alternatives. The alternative that gives a smaller trip completion time is chosen to assign Batch 2. Here, the calculation of trip completion times can be shown as follows: if Batch 2 is assigned to Plant 1, the completion time of the second trip is $\max\left(\sum_{j=1}^{2K} p_j, \max\left(\sum_{j=1}^{K} p_j, d_1\right) + 2d_1\right) + d_1$. The latter term in the maximum is the time truck arrives at Plant 1 after the first trip. If Batch 2 is assigned to Plant 2, the completion time of the second trip is $\max\left(\sum_{j=K+1}^{2K} p_j, \max\left(\sum_{j=1}^{K} p_j, d_1\right) + d_1 + d_2\right) + d_2$. Suppose that Plant 2 was chosen. Then for the third trip, among plant set {Plant 3, …, Plant m}, Plant 3 will be chosen as a potential plant. And between Plants 1 and 2, Plant 1 will be chosen since the sum of the processing times of the jobs already assigned to Plant 1 is smaller than that of those assigned to Plant 2. Hence, the potential plants for the third trip will be Plants 1 and 3. The plant having the smaller trip completion time will be chosen and this procedure will continue until all batches are assigned to some plant. So we can generalize this procedure as follows:

To assign Batch k, among the plants having the same number of batches that have already been assigned to them, the plant having the smallest distance will be chosen and marked as a potential plant. Among all potential plants, the plant giving the smallest possible trip completion time is chosen to assign Batch k.

Having clarified main ideas used, Heuristic 1 can be outlined as follows:

Step 0) Index jobs in non-decreasing order of processing times and plants in non-decreasing order of distances to the customer area. Set total completion time to 0. Go to step 1.

Step 1) Assign the first batch and make the first trip to Plant 1. Update the total completion time. If all jobs are assigned to a plant, stop. Otherwise, go to Step 2.

Step 2) Determine the potential plants. Go to step 3.

Step 3) For the first unassigned batch, calculate possible trip completion times of the potential plants. Go to step 4.

Step 4) Choose the plant with the smallest possible trip completion time. Go to step 5.

Step 5) Assign the first unassigned batch to this plant. Update the total completion time. If there exists any unassigned batch, go to Step 2. Otherwise, stop and return the total completion time.

As can be seen in the pseudocode given in Appendix B.1, Heuristic 1 runs in $O(m^2n + mn^2)$ time. This is polynomial when m is fixed.

Batching the jobs before determining the route of the truck as in Heuristic 1 makes the solution methodology easy to apply in real life. However, we had the intuition that having the flexibility of rearranging the batches, as opposed to the fixed batches in Heuristic 1, may provide us with better results in terms of total completion time. Therefore, we develop another heuristic called Heuristic 2.

Heuristic 2 (LPH2):

In this method, fixed batches do not exist. Assuming trip k visits plant i, and it is required to transport K jobs, if we can produce K jobs on plant i until the truck arrives there, we choose the batch with largest processing time to be assigned to that plant. The

procedure that chooses the batch with largest processing time is called *batch sliding*. The intuition of *batch sliding* comes from the two-dimensional bin packing problem. Each trip in our problem is similar to a bin. First dimension is the number of jobs that needs to be transported on that trip and the second dimension is the available processing time of the plant to which the trip is made. The available processing time of a plant is the time at which the truck arrives at that plant less the time the last job at that plant completes its processing. The main idea here is to use the jobs that can fill as much of the second dimension as possible while using the first dimension completely. This means, among the batches that can fill the first dimension and without exceeding the second dimension, we should choose the batch with the largest processing time, where the processing time of a batch is the sum of the processing times of the jobs in that batch. Since *batch sliding* is also used in some of the methods for other cases, we first illustrate it on an example and then give a step-by-step algorithm.

Example 4.1:

Suppose on the first trip the truck visits Plant i and $K = 2$. The distance of Plant i to the customer is $d_i = 24$. We have a total of 10 jobs with processing times $p = \{p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9, p_{10}\} = \{5,6,8,8,10,11,14,14,15,15\}$. Truck arrives at Plant i at time 24.

- Choose the two jobs with smallest processing times. We can show this batch choice as $\{\boxed{5},\boxed{6},8,8,10,11,14,14,15,15\}$. Since $5+6=11<24$, we can produce two jobs until the truck arrives there. Therefore, we can apply *batch sliding.*
- Sliding the batch one step we will have $\{5,\boxed{6},\boxed{8},8,10,11,14,14,15,15\}$ and since $6+8=14<24$, we will continue sliding until we find the two-size batch with largest processing time that does not violate 24 unit limit.

- For this trip our batch choice will be $\{5,6,8,8,\boxed{10},\boxed{11},14,14,15,15\}$ with a processing time of 21. This batch will be assigned to and transported from Plant i.

Now suppose that we have the following conditions: the truck will visit a plant with an available processing time of 23 and $K=2$. The job set is $\{5,6,8,8,\sout{10},\sout{11},14,14,15,15\}$ where the jobs with processing times 10 and 11 are already assigned to some plant.

- Choose the two jobs with the smallest processing times, shown as $\{\boxed{5},\boxed{6},8,8,\sout{10},\sout{11},14,14,15,15\}$. Since $5+6=11<23$, we can produce two jobs until the truck arrives there. Therefore, we can apply *batch sliding.*
- Sliding the batch as in the previous example we will obtain the following batch choice $\{5,6,8,\boxed{8},\sout{10},\sout{11},\boxed{14},14,15,15\}$ with a processing time of $8+14=22$.

Algorithm Batch Sliding can be summarized as follows:

Step 0) Set $j=1$. Go to step 1.

Step 1) Calculate the sum of processing times of the first K unassigned jobs starting with the job at j^{th} position. Go to step 2.

Step 2) If the sum in step 1 is less than or equal to the available processing time of the plant, increment j and go to step 1. Otherwise, stop and return the batch starting with the job at $(j-1)^{th}$ position.

We can outline Heuristic 2 as follows:

Step 0) Index jobs in non-decreasing order of processing times and plants in non-decreasing order of distances to the customer area. Set total completion time to 0. Set $i = 1$, $k = 0$ and $q = \left\lfloor \frac{n}{K} \right\rfloor$. Go to step 1.

Step 1) Calculate the sum of processing times of the first K unassigned jobs. If the sum is less than or equal to the available processing time of plant 1, apply Algorithm Batch Sliding and go to step 6. Otherwise, go to step 2.

Step 2) Assign the first K unassigned jobs to plant 1, update total completion time and increment k. If $k < q$, go to step 3. Otherwise, go to step 7.

Step 3) Calculate the sum of processing times of the first K unassigned jobs. If the sum is less than or equal to the available processing time of plant i, apply Algorithm Batch Sliding and go to step 6. Otherwise, go to step 4.

Step 4) If $i < m$, increment i and go to step 1. Otherwise, go to step 5.

Step 5) Calculate possible trip completion times with respect to the sum of processing times found in step 1. Choose the plant with smallest trip completion time and assign the first K unassigned jobs to that plant. Update total completion time, set $i = 1$ and increment k. If $k < q$, go to step 3. Otherwise, go to step 7.

Step 6) Assign the batch found by the Batch Sliding Algorithm applied in step 1 to plant i, set $i = 1$, update total completion time and increment k. If $k < q$, go to step 3. Otherwise, go to step 7.

Step 7) If a partially loaded trip is not required, return the total completion time and stop. Otherwise, go to step 8.

Step 8) Calculate the sum of processing times of all unassigned jobs. Calculate possible trip completion times with respect to this sum. Choose the plant with smallest trip

completion time and assign all unassigned jobs to that plant. Update and return the total completion time, and stop.

The pesudocodes of the batch sliding algorithm and Heuristic 2 are given in Appendix B.2. Batch sliding algorithm runs in $O(n^2)$ time and Heuristic 2 runs in $O(mn^3)$ time.

4.2.2.3. Lower Bound for Last Trip Partial Problem (Lower Bound 1)

Although the MIP model presented in Section 4.2.2.1 may give optimal solutions in a reasonable amount of time for small instances, obtaining a result for larger instances requires long CPU times. Therefore, in order to be able to measure the performance of our heuristic methods even with large instances, we develop a lower bound, named as Lower Bound 1. The calculation of the lower bound for this case is mainly based on two parts: the time truck arrives at the customer after a trip and the number of jobs transported in that trip. In this case, the second part is known so the problem here is to determine the first part. Since we do not know the route of the truck, in order to be able to calculate a valid lower bound, all plants are assumed to have the same distance d to the customer, where $d = \min_i d_i$. On the first trip, truck will arrive at a plant at time d and will leave the plant either as soon as it arrives or at the completion time of the K^{th} job. Therefore, we can say the truck leaves the plant on its first trip at time $\max\left(\sum_{j=1}^{K} p_j, d\right)$. On the second trip, the truck visits another plant and to find a valid lower bound, we assume that when it arrives at the plant on its second trip, there are K jobs already produced.

If the plants have the same distance to the customer, the truck visits a different plant in each one of the first m trips. So in the $(m+1)^{th}$ trip, it makes a second visit to a plant, meaning that some jobs have already been assigned to and produced at that plant. Since we

know that only the last trip may be partially loaded, there will be at least K jobs already assigned to the plant to which the $(m+1)^{th}$ trip is made. Therefore, if not completed already until the truck arrives at that plant, the truck has to wait for the completion of $2K^{th}$ job. Hence, for the 1^{st}, $(m+1)^{th}$, $(2m+1)^{th}$, …and so on trips, the departure time of the truck from the plant should be recalculated according to the jobs that have already been assigned to that plant. To clarify, we can divide the trips into *rounds* and since there are m plants, there will be m trips in each *round*. The first m trips will be in *round1*, the second m trips will be in *round2*, and so on. Since $\left\lceil \frac{n}{K} \right\rceil$ trips are required to transport all jobs, there will be $\left\lceil \left\lceil \frac{n}{K} \right\rceil / m \right\rceil$ rounds. In Table 4.1, for each trip on the first three rounds, the calculation of the arrival time of that trip to the customer is given.

Table 4.1: Sample calculation of completion times of the trips for Lower Bound 1

	Trip 1	**Trip 2**	**…**	**Trip** m
Round1	$L_1 + d = \max\left(\sum_{j=1}^{K} p_j, d\right) + d$	$L_1 + 3d$	$\cdots$	$B_1 = L_1 + (2m-1)d$
Round2	$L_2 + d = \max\left(\sum_{j=1}^{2K} p_j, B_1 + d\right) + d$	$L_2 + 3d$	$\cdots$	$B_2 = L_2 + (2m-1)d$
Round3	$L_3 + d = \max\left(\sum_{j=1}^{3K} p_j, B_2 + d\right) + d$	$L_3 + 3d$	$\cdots$	$B_3 = L_3 + (2m-1)d$

As can be seen in Table 4.1, the completion time of a trip is composed of two terms: L_v, where $v = 1, 2, ..., \left\lceil \left\lceil \frac{n}{K} \right\rceil / m \right\rceil$, and rd, where $r = 1, 2, ..., 2m-1$. The term L_v will be called *LeaveTime*, and the term rd will be called *TravelTime*.

Keeping the calculation procedures shown in Table 4.1 in mind, the algorithm of Lower Bound 1 can be outlined as follows:

Step 0) Index jobs in non-decreasing order of processing times and plants in non-decreasing order of distances to the customer area. Set total completion time to 0, $k = 1$ and *LeaveTime* = 0. Set $q = \left\lfloor \frac{n}{K} \right\rfloor$ and $q' = \left\lceil \frac{n}{K} \right\rceil$

Step 1) Calculate *TravelTime* for each trip.

Step 2) If $k \in \{1, m+1, 2m+1, ...\}$, go to step 3. Otherwise, go to step 4.

Step 3) Update *LeaveTime* and go to step 4.

Step 4) Calculate completion time of the trip by summing *LeaveTime* computed in step 3 and *TravelTime* of that trip. Increment k. If $k \leq q'$, go to step 2. Otherwise, go to step 5.

Step 5) If a partially loaded trip is not required, go to step 6. Otherwise, go to step 7.

Step 6) Sum the completion times of each trip and multiply it with K. Set the total completion time to this value. Return the total completion time and stop.

Step 7) Sum the completion times of the first q trips and multiply it with K. Multiply the completion time of the $(q')^{th}$ trip with $(n - qK)$ and add to this sum. Set the total completion time to the value found. Return the total completion time and stop.

The pseudocode of the algorithm of Lower Bound 1 is given in Appendix A.1. This algorithm runs in $O(n^2 + mn)$ time.

We also generated a two-stage lower bound where the plants are at the same location (with the distance of the closest plant) and parallel. The jobs are first assigned to the plants in SPT order and then transferred to the customer. However, this lower bound was outperformed by Lower Bound 1 in our preliminary tests.

4.2.3. Exact Solution Method for Fully Loaded Trips Problem

In this problem it is not known which trip will be partially loaded; hence, the number of jobs transported in each trip is a variable. Therefore, trying to calculate the total completion time by multiplying the completion time of each trip with the number of jobs transported on that trip would cause nonlinearity. Thus, the total completion time will be calculated by summing up the completion times of all jobs, where the completion time of a job is defined as the time it reaches at the customer.

Parameters:

$d_{\max}$: $\max_i d_i$

Assumptions:

We assume without loss of generality that jobs are indexed in non-decreasing order of their processing times. That is, $p_1 \leq p_2 \leq .... \leq p_n$.

Decision Variables:

$$X_{ij} = \begin{cases} 1, & \text{if job } j \text{ is assigned to plant } i \\ 0, & \text{otherwise} \end{cases} \quad \text{for } i = 1,2,....,m \text{ and } j = 1,2,....,n$$

$$Y_{jk} = \begin{cases} 1, & \text{if job } j \text{ is transported on trip } k \\ 0, & \text{otherwise} \end{cases} \quad \text{for } j = 1,2,....,n \text{ and } k = 1,2,....,q'$$

$$W_{ik} = \begin{cases} 1, & \text{if plant } i \text{ is visited on trip } k \\ 0, & \text{otherwise} \end{cases} \quad \text{for } i = 1,2,....,m \text{ and } k = 1,2,....,q'$$

$$F_k = \begin{cases} 1, & \text{if the truck is fully loaded on trip } k \\ 0, & \text{otherwise} \end{cases} \quad \text{for } k = 1,2,....,q'$$

C_j = process completion time of job j at the plant it is assigned to, $j = 1,2,....,n$

Q_k = time truck reaches the customer at the end of the k^{th} trip, $k = 1,2,....,q'$

T_j = completion time of job j, i.e. the time that job reaches the customer, $j = 1,2,....,n$

Mixed Integer Programming Model (MIP-2)

(17) minimize $\sum_{j=1}^{n} T_j$

subject to

(2), (3), (6) – (16)

(18) $T_j \geq Q_k - M_k(1-Y_{jk}), \qquad j=1,..,n\ , \quad k=1,..,q'$

(19) $\sum_{j=1}^{n} Y_{jk} \leq K\ , \qquad k=1,..,q'$

(20) $K - \sum_{j=1}^{n} Y_{jk} \leq \left[1-F_k\right]\cdot K, \qquad k=1,..,q'$

(21) $\sum_{k=1}^{q'} F_k = q'-1$

(22) T_j *is nonnegative real number*, $\qquad j=1,...,n$

(23) $F_k \in \{0,1\}, \qquad k=1,...,q'$

Constraint set (18) ensures that if job j is transported on trip k, its completion time cannot be smaller than the completion time of trip k, i.e. the time the truck reaches at the customer at the end of the k^{th} trip. M_k in constraint set (18) is a large integer and its value is determined to be $k\cdot\left(\sum_{j=1}^{n} p_j + 2d_{\max}\right)$, for $k=1,...,q'$. Constraint set (19) implies that on each trip, at most K jobs can be transported. Constraint set (20) and constraint (21) work together. When there is a partially loaded trip, its F_k value is forced to be 0 by constraint set (20), while constraint (21) ensures that there can be exactly one partially loaded trip. If a partially loaded trip is not required, constraint (21) will again assign the value of 0 to one of the F_k values. However, since having an F_k value equal to 0 in constraint set (20), does not prevent the trip from being fully loaded, we will still have all trips fully loaded.

4.2.4. Heuristic Methods for Fully Loaded Trips Problem

The two heuristic methods proposed for this problem are generalizations of the heuristics developed for the problem with the constraint that the partial trip, if required, will be the last one.

Heuristic 3 (FTH1):

This method is a generalization of Heuristic 1. The main idea is to calculate the total completion times of q' alternatives, where $q' = \left\lceil \frac{n}{K} \right\rceil$ and k^{th} alternative assumes that k^{th} trip is partially loaded. Among these alternatives, the one yielding the smallest total completion time is chosen. The completion time of each alternative can be calculated by using a slightly modified version of Heuristic 1. A detailed pseudocode of Heuristic 3 is given in Appendix B.3. This heuristic runs in $O(mn^3)$ time which is polynomial when m is fixed.

In this method the two principles introduced for Heuristic 1, batching the jobs and choosing the potential plants, are used. Below is an example showing how to batch the jobs for different alternatives supposing that a partially loaded trip is required.

1st Trip Partial: $\underbrace{J_1, J_2, ..., J_{n-qK}}_{Batch1(size\ n-qK)}, \underbrace{J_{n-qK+1}, J_{n-qK+2}, ..., J_{n-qK+K}}_{Batch2}, ..., \underbrace{J_{n-K+1}, J_{n-K+2}, ..., J_n}_{Batchq'}$

2nd Trip Partial: $\underbrace{J_1, J_2, ..., J_K}_{Batch1}, \underbrace{J_{K+1}, J_{K+2}, ..., J_{n-qK+K}}_{Batch2(size\ n-qK)}, ..., \underbrace{J_{n-K+1}, J_{n-K+2}, ..., J_n}_{Batchq'}$

…….

$(q')^{th}$ Trip Partial: $\underbrace{J_1, J_2, ..., J_K}_{Batch1}, \underbrace{J_{K+1}, J_{K+2}, ..., J_{2K}}_{Batch2}, ..., \underbrace{J_{qK+1}, J_{qK+2}, ..., J_n}_{Batchq'(size\ n-qK)}$

Heuristic 4 (FTH2):

This heuristic is a generalization of Heuristic 2. The batch sliding mechanism introduced for Heuristic 2 is also implemented in Heuristic 4. As in Heuristic 3, q' alternatives are compared to each other and the one with the smallest total completion time is chosen. The completion time of each alternative can be calculated by using a slightly modified version of Heuristic 2. This algorithm runs in $O(mn^4)$ time and its pseudocode can be found in Appendix B.4.

4.2.5. Lower Bounds for Fully Loaded Trips Problem (Lower Bounds 2, 3 and 4)

In order to be able to assess the performance of proposed heuristics in large instances, we develop two fast lower bounds, referred to as Lower Bound 2 and Lower Bound 3. In our preliminary studies, we tried solving the MIP model developed for this problem with equal distances d, where $d = \min_i d_i$, in order to obtain a lower bound. However, even with small instances, it takes considerable amount of CPU time to obtain optimal solutions.

Lower Bound 2:

In this problem at most one trip is partially loaded. It is required to have a total of q' trips, where $q' = \left\lceil \frac{n}{K} \right\rceil$, to transport all jobs. q of these trips will be fully loaded, where $q = \left\lfloor \frac{n}{K} \right\rfloor$. The number of jobs to be carried on the partial trip is $(n - qK)$.

The total completion time of a trip consists of two elements: the number of jobs transported in that trip and the time the truck arrives at the customer at the end of that trip.

Number of jobs transported on the trips: We consider as if whenever the truck arrives at a plant, there will be K jobs already produced at that plant. Hence, for the first q trips we suppose that K jobs are transported on that trip and $(n-qK)$ jobs are transported on the last trip.

Trip completion times: This method simplifies the problem by setting the distance from each plant to the customer equal to the smallest such distance. Denote this identical distance between a plant and the customer location by d. While determining the number of jobs transported on the trips we supposed that the first trip will carry K jobs. However, in reality, the first trip may be the one that is partially loaded. Hence, in our lower bound calculations, if we state that the truck waits for the completion time of the K^{th} job in the first trip, the lower bound obtained would not be valid. In order to alleviate this issue and obtain a valid lower bound, we suppose that the truck leaves the plant visited in the first trip either as soon as it arrives or at the completion time of the $(n-qK)^{th}$ job. Trip completion times and the number of jobs carried on those trips can be calculated as shown in Table 4.2.

Hence, if a partially loaded trip is not required, the lower bound is obtained using the following formula:

$$K\cdot\left[A+d\right]+K\cdot\left[A+3d\right]+....+K\cdot\left[A+(2\cdot\frac{n}{K}-1)d\right]=n\cdot A+K\cdot d\cdot q^2$$

where $A=\max\left(\sum_{j=1}^{K}p_j,d\right)$.

If a partially loaded trip is required, the lower bound can be found using the following formula:

$$K\cdot\left[B+d\right]+K\cdot\left[B+3d\right]+....+K\cdot\left[B+(2q-1)d\right]+(n-qK)\cdot\left[B+(2q+1)d\right]$$

$$=n\cdot B+K\cdot d\cdot q^2+(n-q\cdot K)\cdot(2q+1)\cdot d \text{ where } B=\max\left(\sum_{j=1}^{n-qK}p_j,d\right).$$

Table 4.2 : Trip Completion Times and Number of Jobs Carried in Lower Bound 2

	Partially Loaded Trip Not Required		**Partially Loaded Trip Required**	
	Trip Arrival Time	Number of Jobs Carried	Trip Arrival Time	Number of Jobs Carried
Trip 1	$A+d=\max\left(\sum_{j=1}^{K} p_j, d\right)+d$	K	$B+d=\max\left(\sum_{j=1}^{n-qK} p_j, d\right)+d$	K
Trip2	$A+3d$	K	$B+3d$	K
. . .	. . .	. . .	. . .	. . .
Trip q	$A+(2q-1)d$	K	$B+(2q-1)d$	K
Trip q'	---	---	$B+(2q+1)d$	$n-qK$

Lower Bound 2 may be very loose when the processing times of the jobs are large so that the truck may not be filled (at the time it arrives at a plant) in all trips. To address this situation, we develop Lower Bound 3 as an alternative procedure.

Lower Bound 3:

Once again, the total completion time of a trip is mainly based on two parts: the time the truck arrives at the customer at the end of that trip and the number of jobs transported in that trip.

Number of jobs transported on the trips: To obtain a valid lower bound, we consider as if the truck carries K jobs in its first q trips and $(n-qK)$ jobs on the last trip as explained in Lower Bound 2.

Trip completion times: Here, in the calculation of trip completion times, we follow the same updating mechanism used in the first trips of the rounds in Lower Bound 1. We assume the plants have the same distance d, where $d = \min_i d_i$, to the customer area. Since the plants have the same distance, the truck visits a different plant in each one of the first m trips. So in the $(m+1)^{th}$ trip, it makes a second visit to a plant, meaning that some jobs have already been assigned to and produced at that plant. There will be at least $(n-qK)$ jobs already assigned to the plant to which the $(m+1)^{th}$ trip is made. Therefore, if not completed already until the truck arrives at that plant, the truck has to wait for the completion of the $(n-qK+K)^{th}$ job. Hence, for the 1^{st}, $(m+1)^{th}$, $(2m+1)^{th}$, …and so on trips, the departure time of the truck from the plant should be recalculated according to the jobs that have already been assigned to that plant.

In Table 4.3, the completion times of all the trips until the end of round 3 are given. The completion times of the trips for other rounds can be calculated by following the procedure shown in this table.

Table 4.3 : Trip Completion Times for Lower Bound 3

	Trip 1	**Trip 2**	**…**	**Trip** m
Round1	$L_1 + d = \max\left(\sum_{j=1}^{n-qK} p_j, d\right) + d$	$L_1 + 3d$		$B_1 = L_1 + (2m-1)d$
Round2	$L_2 + d = \max\left(\sum_{j=1}^{n-qK+K} p_j, B_1 + d\right) + d$	$L_2 + 3d$		$B_2 = L_2 + (2m-1)d$
Round3	$L_3 + d = \max\left(\sum_{j=1}^{n-qK+2K} p_j, B_2 + d\right) + d$	$L_3 + 3d$		$B_3 = L_3 + (2m-1)d$

Lower Bound 3 uses the same step-by-step procedure with Lower Bound 1. However, in step 3, updating *LeaveTime* is done according to the calculation methods shown in Table 4.3, instead of Table 4.1. The pseudocode of the algorithm of Lower Bound 3 is given in Appendix A.2. This algorithm runs in $O(mn+n^2)$ time.

In Lower Bound 3, as mentioned above, the departure time of the first trip from the plant is $\max\left(\sum_{j=1}^{n-qK} p_j, d\right)$. Hence, as $(n-qK)$ gets smaller, the departure time of the first trip may get smaller. Especially for the problems considering jobs with large processing times, this may cause the lower bound to be loose. To overcome this problem we develop an alternative lower bound procedure.

Lower Bound 4:

In this lower bound generation method, our aim is to eliminate the partially loaded trip. In order to do this, from the job set, we delete the last $(n-qK)$ jobs. Then we apply exactly the same procedure as the one used in Lower Bound 3. But since we deleted the last $(n-qK)$ jobs, there will occur only fully loaded trips and the first trip leaves the plant at time $\max\left(\sum_{j=1}^{K} p_j, d\right)$.

4.3. General Problem

In the previous problems, we considered a constraint such that the truck will have a full load in all trips, except one. The problem considered in this section is more general in the sense that it does not involve any such constraint. As explained in Section 4.2, fully loaded trips constraint may be preferred by the companies producing physically sensitive products.

The solution to this more general problem may be applied by the companies that produce less sensitive products with more durable packages so that product would not be harmed when shaken or scattered around in the truck.

In most applications, when the truck arrives at a plant, it takes the products ready for delivery and turns back to the customer. This is due to the dispatcher's false belief that doing so would result in a better completion time performance on the average. In order to address this common practical application, we study the problem where the truck leaves the plant immediately. Before continuing with the proposed methods to solve this problem, we will first prove in Theorem 4 that the main problem defined in Chapter 3 is at least NP-hard in the ordinary sense. In Corollary 2, it will be shown that the problem with no wait constraint is at least NP-hard in the ordinary sense.

4.3.1. NP-Hardness Proofs for the General Problem and the Problem with No Wait Constraint

Lee and Chen (2001), show that the problem $P2 \to 1|v=1, K \geq 1|\sum C_j$ is NP-hard. Although this problem is different from ours, their proof shows that our problem is NP-hard. However, we believe that our proof is more intuitive and easier to understand in the context of the main problem studied in this thesis.

Theorem 3: Problem $2P \to 1|v=1, K \geq 1|\sum C_j$ is NP-hard.

Proof: Given an instance of the PARTITION problem, we construct the following instance of our problem.

Number of jobs $n = 4h$, and jobs set $N = H \cup \{2h+1\} \cup \{2h+2, 2h+3, ..., 4h\}$ where $H = \{1, 2, ..., 2h\}$

Number of plants $m = 2$

Truck capacity $K = 2h$

Processing times $p_j = s\left(a_j\right)$ for $j \in H$, $p_{2h+1} = \sum_{j \in H} p_j = 2B$,

$$p_j = 0 \text{ for } j = \{2h+2, 2h+3, ..., 4h\}$$

Plant distances $d_1 = d_2 = d = \frac{\sum_{j \in H} p_j}{2} = B$

Threshold on total completion time $C = 12Bh$

We first prove that the total completion time will always be greater than C if more than two trips are used. It is obvious that at least two trips are required to transport all jobs. If exactly two trips are used, K jobs will be transported in each trip. Assume that in the first trip, instead of sending K jobs, $K - X_1$ jobs were sent, and in the second trip, $K - X_2$ jobs were sent, where $X_1 + X_2 > 0$ and $X_1, X_2 \in$ □. Assuming Y_1 jobs were transported in the third trip, Y_2 jobs in the fourth trip, and Y_Q jobs in the $(Q+2)^{nd}$ trip, such that $\sum_{l=1}^{Q} Y_l = X_1 + X_2$ and $Y_{l_1} \geq Y_{l_2}$ for every $l_1 < l_2$ (i.e. on later trips fewer jobs will be sent), we will obtain a total completion time as follows:

$$\sum C_j \geq 2(K - X_1)d + 4(K - X_2)d + 6Y_1 d + 8Y_2 d + ... + 2Y_Q(Q+2)d$$

$$\geq 6Kd - 2dX_1 - 4dX_2 + 6d(Y_1 + Y_2 + ... + Y_Q)$$

$$= 6dK + 4dX_1 + 2dX_2 > C$$

Since using more than two trips results in a total completion time greater than C, exactly two trips must be used to transport all the jobs. Therefore, proof of Theorem 2 can be used to prove the NP-hardness of this problem.■

The following corollary immediately follows.

Corollary 3: The problem $MP \to 1|v=1, K \geq 1|\sum C_j$ is NP-hard.

Proof: Since $2P \to 1|v=1, K \geq 1|\sum C_j$ is NP-hard, $MP \to 1|v=1, K \geq 1|\sum C_j$ is also NP-hard.

In this section, along with the general problem, we consider one of its special cases, the problem with no wait constraint. This problem can be denoted as $MP \to 1|v=1, K \geq 1, NoWait|\sum C_j$.

Corollary 4: Problem $2P \to 1|v=1, K \geq 1, NoWait|\sum C_j$ with no wait constraint is NP-hard.

Proof:

Since the truck does not wait in Theorem 4, it is easy to see that the proof of Theorem 4 is also valid for this problem.

4.3.2. The Problem with No Wait Constraint

As mentioned before, in this problem, we have the no wait constraint such that when the truck arrives at a plant, it takes the jobs already produced there without violating the capacity constraint and leaves the plant as soon as it arrives there.

4.3.2.1. Exact Solution Method for the Problem with No Wait Constraint

In this problem, since we do not know how many trips are necessary to transport all jobs, we try to cover the worst case, i.e. the case each job is transported on separate trips, by having n trips. Therefore, we have $k = 1, 2, ..., n$ trips.

Assumptions:

We assume without loss of generality that jobs are indexed in non-decreasing order of their processing times. That is, $p_1 \le p_2 \le .... \le p_n$.

Decision Variables:

$$X_{ij} = \begin{cases} 1, & \text{if job } j \text{ is assigned to plant } i \\ 0, & \text{otherwise} \end{cases} \quad \text{for } i = 1,2,....,m \text{ and } j = 1,2,....,n$$

$$Y_{jk} = \begin{cases} 1, & \text{if job } j \text{ is transported on trip } k \\ 0, & \text{otherwise} \end{cases} \quad \text{for } j = 1,2,....,n \text{ and } k = 1,2,....,n$$

$$W_{ik} = \begin{cases} 1, & \text{if plant } i \text{ is visited on trip } k \\ 0, & \text{otherwise} \end{cases} \quad \text{for } i = 1,2,....,m \text{ and } k = 1,2,....,n$$

C_j = process completion time of job j at the plant it is assigned to, $j = 1,2,....,n$

T_j = time job j reaches the customer, $j = 1,2,....,n$

Q_k = time truck reaches the customer at the end of the k^{th} trip, $k = 1,2,....,n$

Mixed Integer Programming Model (MIP-3)

(17) minimize $\sum_{j=1}^{n} T_j$

subject to

(2), (3), (8), (12), (14), (22)

(18.2) $T_j \geq Q_k - M_k(1-Y_{jk}), \qquad j=1,..,n\ , \quad k=1,..,n$

(19.2) $\sum_{j=1}^{n} Y_{jk} \leq K\ , \qquad k=1,..,n$

(6.2) $W_{ik} \geq X_{ij} + Y_{jk} - 1, \qquad i=1,..,m\ , \quad j=1,..,n\ , \quad k=1,..,n$

(7.2) $\sum_{i=1}^{m} W_{ik} \leq 1, \qquad k=1,..,n$

(9.2) $Q_1 = 2 \cdot \sum_{i=1}^{m} (W_{i1} \cdot d_i)$

(10.2) $Q_k \geq Q_{k-1} + 2 \cdot \sum_{i=1}^{m} (W_{ik} \cdot d_i), \qquad k=2,...,n$

(11.2) $(M_1 + d_i) \cdot (2 - X_{ij} - Y_{jk}) + Q_{k-1} + \sum_{v=1}^{m} (W_{vk} \cdot d_v) \geq C_j, \qquad i=1,...,m\ , \quad j=1,...,n\ ,$

$k=1,...,n$

(13.2) Q_k *is nonnegative real number*, $\qquad k=1,...,n$

(15.2) $Y_{jk} \in \{0,1\}, \qquad j=1,...,n\ , \quad k=1,...,n$

(16.2) $W_{ik} \in \{0,1\}, \qquad i=1,...,m\ , \quad k=1,...,n$

Most of the constraints of the form ($a.2$) are the modifications of the constraints of the form (a) with a difference in just the condition for the trip number, k . In constraint set (7.2), the equality form of constraint set (7) is changed into an inequality since we have $W_{ik} = 0$ for the trips that may not be used. Conversely in constraint (9.2), the inequality form of constraint (9) is changed into equality because we know that the truck does not wait at the plant and leaves immediately as it arrives there. Since in this constraint only the first trip is considered, the trip is completed at time $2d_i$, if the truck visits plant i . The model will still be valid if the inequality in constraint set (10.2) is changed into equality. Although not

intuitive, preliminary tests indicated that having equality in this constraint results in larger CPU times. Thus, we use inequality. Constraint set (11.2) ensures that if a job is going to be transported in a trip, it should complete its processing on or before the truck arrives at the plant at which it is processed.

In addition to the above constraints, we also have the following valid inequality:

$$\sum_{i=1}^{m} W_{ik} \leq \sum_{i=1}^{m} W_{i(k-1)}, \qquad k = 2,..,n$$

This inequality implies that if trip k is not used, then trip $k+1$ will not be used. Actually, this inequality adds a cut to our model.

4.3.2.2. Heuristic Methods for the Problem with No Wait Constraint

Our preliminary tests show that it is not possible to obtain optimal solutions in a reasonable amount of time using the MIP model presented in Section 4.3.2.1. Therefore, we develop two heuristic methods to generate near optimal solutions for this problem.

Heuristic 5 (NWH1):

Since our objective is to minimize the total completion time, we want to transport more jobs in the beginning of the schedule. However, it is important not to increase the completion times of the trips excessively while trying to increase the number of jobs transported. For example, by visiting a farther plant in a trip, it is possible to transport more jobs in that and subsequent trips, but this may also increase the completion times of these trips. Therefore, there is a tradeoff between the number of jobs transported and the completion times of the trips. The main idea of this method arises from the desire to balance this tradeoff.

First, we define the gain of going to a farther plant as the additional number of jobs that can be transported by going to that farther plant instead of a closer plant. The number of jobs to be transported from a plant is determined by taking the minimum of the number of jobs that can be produced in the available processing time of that plant and the capacity of the truck. As mentioned in Heuristic 2 in Section 4.2.2.2, the available processing time of a plant is the time at which the truck arrives at that plant minus the time the last job assigned to that plant completes its processing.

Secondly, we define the loss as the delay in the completion time of the trip by going to a farther plant instead of going to the closest plant. Although the gain is subject to change depending on the processing times of the jobs that have not been transported yet, the loss is constant for each plant. We can calculate the loss of going to plant i instead of the closest plant as $2(d_i - d)$, where $d = \min_i d_i$.

In this method, we try to balance the tradeoff between gain and loss, by visiting a farther plant only if the gain of this trip justifies the loss. The gain-loss justification mechanism depends on two conditions:

1. The gain, denoted as g, should be larger than the loss divided by $\bar{p}$, where $\bar{p} = \frac{1}{n} \cdot \sum_{j=1}^{n} p_j$. In mathematical representation that is, $g > \frac{2(d_i - d)}{\bar{p}}$.
2. The gain should be greater than or equal to 1.

If both of the above conditions are satisfied, then Heuristic 5 lets the truck visit a farther plant instead of the closest one.

In the first condition, the ratio $\frac{2(d_i - d)}{\bar{p}}$ is determined as a result of preliminary experiments. We expressed the ratio as $\frac{a \cdot 2(d_i - d)}{4 \cdot \bar{p}}$, and tested its performance on some test

problems for different a values, where $a = 1, 2, ..., 8$. $a = 4$ is found to perform better than the others from the perspective of minimizing the total completion time.

In this method, in each trip of the truck, the unassigned jobs with smallest processing times will be assigned to a plant and transported. This means, the batch sliding mechanism explained in Heuristic 2 is not implemented.

Following the discussions above, Heuristic 5 can be summarized as follows:

Step 0) Index jobs in non-decreasing order of processing times and plants in non-decreasing order of distances to the customer area. Set total completion time to 0, $numSent = 0$. Calculate $\bar{p}$. For each plant i, set $numJobs[i] = 0$.

Step 1) Set $i = 1$ and calculate the number of jobs that can be produced in the available processing time of plant i. If the value found is greater than or equal to K, set $numJobs[i] = K$ and go to step 2. Otherwise, go to step 3.

Step 2) Assign the first $numJobs[i]$ unassigned jobs to plant i. Increase $numSent$ by $numJobs[i]$. Update the total completion time. If $numSent < n$, go to step 1. Otherwise, return the total completion time and stop.

Step 3) Increment i and calculate the number of jobs that can be produced in the available processing time of plant i. If the value found is greater than or equal to K, set $numJobs[i] = K$ and go to step 4. Otherwise, go to step 4 without changing the value of $numJobs[i]$.

Step 4) If $numJobs[i]$ satisfies *gain-loss justification* conditions, go to step 2. Otherwise, go to step 5.

Step 5) If $i < m$, go to step 3. Otherwise, set $i = 1$ and go to step 2.

Heuristic 5 runs in $O(mn^2)$ time and its detailed pseudocode can be found in Appendix B.5.

Heuristic 6 (NWH2):

In this method, we employ the batch sliding mechanism introduced in Heuristic 2. The main idea is as follows: when the truck makes a visit to a plant, the number of jobs it can produce in the available processing time of that plant, called *numJobs*, is calculated. Then, procedure batch sliding is applied to select the batch with largest processing time among the batches having *numJobs* jobs. The difference between Heuristic 5 and this heuristic is in step 2. In this method, in step 2, instead of directly assigning the first $numJobs[i]$ unassigned jobs to Plant i, the batch determined as a result of batch sliding is assigned to plant i. The pseudocode of this algorithm is given in Appendix B.6. This algorithm runs in $O(mn^3)$ time.

4.3.2.3. Lower Bound for the Problem with No Wait Constraint (Lower Bound 5)

As mentioned before, in this problem we assume that the truck will leave the plant immediately after it arrives there and take the jobs ready there. Therefore, the departure time of the truck from a plant is determined if the arrival time of it to that plant is known. However, the arrival time of the truck to a plant depends on the route of the truck and the route is not known.

In this method, the total completion time is again composed of two parts: trip completion times and the number of jobs transported on each trip. Since the route of the truck is not known, to find a valid lower bound, we will treat as if the k^{th} trip is completed at time

$k \cdot 2d$, where $d = \min_i d_i$. Determining the completion time of each trip, the problem now is to find out how many jobs will be transported on each trip.

Since we set the trip completion times to their respective minimums, in order to find a valid lower bound, we should set the number of jobs to be transported on each trip to their respective maximums. That means, maximum number of jobs are transported in minimum time. For this purpose, this method tries to determine at most how many jobs can be loaded to the truck in each trip and hence how many trips would the truck make in order to transport all jobs.

Although explained in Section 4.2.2.3, it is better to remind the concept of *rounds*. The trips are grouped into *rounds* and since there are m plants, there exist m trips in each *round*. In other words, a *round* is completed when each plant is visited once.

On the first trip of the first round, the maximum number of jobs would be transported if the farthermost plant is visited. Since it takes the truck d_m units of time to go there, at most l_1 jobs can be produced until the arrival of the truck, where $\sum_{j=1}^{l_1} p_j \le d_m < \sum_{j=1}^{l_1+1} p_j$. In order not to violate capacity constraints, at most $\min(l_1, K)$ jobs can be transported. Then on the second trip, we can transport the greatest number of jobs from Plant $m-1$. The truck arrives there at time $2d_m + d_{m-1}$. At most l_2 jobs can be produced until the arrival of the truck, where $\sum_{j=1}^{l_2} p_j \le 2d_m + d_{m-1} < \sum_{j=1}^{l_2+1} p_j$, and at most $\min(l_2, K)$ number of jobs are transported. Calculation goes on like this until the m^{th} trip. On the m^{th} trip, the truck arrives at the plant at time $t = 2d_m + 2d_{m-1} + ... + 2d_2 + d_1$ and at most l_m jobs can be produced until that time, where $\sum_{j=1}^{l_m} p_j \le t < \sum_{j=1}^{l_m+1} p_j$.

On the second round, i.e. the round starting with $(m+1)^{th}$ trip, on each plant visited, there will be some jobs already assigned to and produced at that plant. Since our policy is to calculate the maximum number of jobs that can be transported on each trip, we want to find the maximum number of jobs that can be produced on each plant until the truck arrives there. Therefore, for each plant, we suppose the minimum number of jobs have already been produced there. For the trips between $(m+1)^{th}$ and $(2m)^{th}$, this minimum number of jobs is equal to $l_{\min}$, where $\sum_{j=1}^{l_{\min}} p_j \le d_1 < \sum_{j=1}^{l_{\min}+1} p_j$. However, among these produced jobs, $\min(l_{\min}, K)$ number of jobs have been already transported. Then, for the $(m+1)^{th}$ trip, to transport the maximum number of jobs, the truck goes again to the farthermost plant. It arrives there at time $(2d_m + 2d_{m-1} + ... + 2d_2 + 2d_1) + d_m$. Hence, on that plant, at most l_{m+1} jobs can be produced, where $\sum_{j=1}^{l_{m+1}} p_j \le (2d_m + 2d_{m-1} + ... + 2d_2 + 2d_1) + d_m < \sum_{j=1}^{l_{m+1}+1} p_j$. However, at least $\min(l_{\min}, K)$ of them have been already transported. So there are $(l_{m+1} - \min(l_{\min}, K))$ jobs produced and ready for transportation, and $\min(l_{m+1} - \min(l_{\min}, K), K)$ of them are transported on this trip. The calculations are similar for the following trips including the $(2m)^{th}$ trip.

On the first trip of the third round, i.e. $(2m+1)^{th}$ trip, the value of $l_{\min}$ is updated. Now, we suppose that we have already made two trips to the closest plant, therefore, on the second trip to that plant, the truck arrives there at the earliest at time $3d_1$. Hence, for the trips between the $(2m+1)^{th}$ and the $(3m)^{th}$, $l_{\min}$ is updated so that $\sum_{j=1}^{l_{\min}} p_j \le 3d_1 < \sum_{j=1}^{l_{\min}+1} p_j$. And the number of jobs that have already been transported is $\min(l_{\min}, 2K)$. Therefore, on $(2m+1)^{th}$ trip, at most $\min(l_{2m+1} - \min(l_{\min}, 2K), K)$ jobs can be transported. This updating mechanism is repeated on the first trip of each round.

We can summarize the notations and calculations used above as follows. Suppose on the k^{th} trip, the truck arrives at the plant it visits at time t. Denote l_k to be the number of jobs that can be produced on that plant until time t. Assume that the truck's previous visit to that plant was on time $t_{\min}$. Denote $l_{\min}$ to be the number of jobs already produced on that plant, i.e. the number of jobs produced until time $t_{\min}$. Assuming that trip k is on the T^{th} round, we can express the number of jobs to be transported on trip k as $\min(l_k - \min\left(l_{\min}, (T-1)\cdot K\right), K)$. Note that for the trips in the first round $l_{\min} = 0$.

Keeping the calculations shown in Table 4.4 in mind, the algorithm for Lower Bound 5 can be summarized as follows:

Step 0) Index jobs in non-decreasing order of processing times and plants in non-decreasing order of distances to the customer area. Initialize *noJobs* array. Set $k = 1$ and $numSent = 0$.

Step 1) Calculate t and $t_{\min}$ for trip k as shown in Table 4.4. Calculate l_k and $l_{\min}$ with respect to t and $t_{\min}$ values.

Step 2) Set $noJobs[k] = \min(l_k - \min\left(l_{\min}, K\right), K)$. If $noJobs[k] \le n - numSent$, go to step 3. Otherwise, go to step 4.

Step 3) Increase *numSent* by $noJobs[k]$. If $numSent < n$, increment k and go to step 1. Otherwise, i.e. if $numSent = n$, go to step 5.

Step 4) Set $noJobs[k] = n - numSent$. Go to step 5.

Step 5) Set the total completion time to $\sum_{k'=1}^{k}\left[(2k'd)noJobs[k']\right]$. Return the total completion time and stop.

Table 4.4: t and $t_{\min}$ calculation for Lower Bound 5

	Trip Number	t **value**	$t_{\min}$ **value**
Round 1	1	$A_1 = d_m$	0
	2	$A_2 = 2d_m + d_{m-1}$	0
	….	…	…
	m	$A_m = 2d_m + 2d_{m-1} + ... + 2d_2 + d_1$	0
Round 2	$m+1$	$2(d_m + d_{m-1} + ... + d_2) + A_1$	d_1
	$m+2$	$2(d_m + d_{m-1} + ... + d_2) + A_2$	d_1
	…	…	…
	$2m$	$2(d_m + d_{m-1} + ... + d_2) + A_m$	d_1
Round 3	$2m+1$	$4(d_m + d_{m-1} + ... + d_2) + A_1$	$3d_1$
	$2m+2$	$4(d_m + d_{m-1} + ... + d_2) + A_2$	$3d_1$
	…	…	…
	$3m$	$4(d_m + d_{m-1} + ... + d_2) + A_m$	$3d_1$

This lower bound generation algorithm runs in $O(m^2 + n^2)$ time and a pseudocode of it is given in Appendix A.3.

4.3.3. Exact Solution Method for the General Problem

Exact solution method for this problem, is very similar to MIP-3. Only two constraints are different. Instead of constraint (9.2) of MIP-3, this method has constraint (9). And instead of (11.2), this method has constraint (11.3) which has the same inequality with constraint (11) but different domain for k.

Assumptions:

We assume without loss of generality that jobs are indexed in non-decreasing order of their processing times. That is, $p_1 \le p_2 \le .... \le p_n$.

Decision Variables:

$$X_{ij} = \begin{cases} 1, & \text{if job } j \text{ is assigned to plant } i \\ 0, & \text{otherwise} \end{cases} \quad \text{for } i = 1,2,....,m \text{ and } j = 1,2,....,n$$

$$Y_{jk} = \begin{cases} 1, & \text{if job } j \text{ is transported on trip } k \\ 0, & \text{otherwise} \end{cases} \quad \text{for } j = 1,2,....,n \text{ and } k = 1,2,....,n$$

$$W_{ik} = \begin{cases} 1, & \text{if plant } i \text{ is visited on trip } k \\ 0, & \text{otherwise} \end{cases} \quad \text{for } i = 1,2,....,m \text{ and } k = 1,2,....,n$$

C_j = process completion time of job j on the plant it is assigned to, $j = 1,2,....,n$

T_j = time job j reaches the customer, $j = 1,2,....,n$

Q_k = time truck reaches the customer at the end of k^{th} trip, $k = 1,2,....,n$

Mixed Integer Programming Model (MIP-4)

(17) minimize $\sum_{j=1}^{n} T_j$

subject to

(2), (3), (6.2), (7.2), (8), (9), (10.2), (12), (13.2), (14), (15.2), (16.2), (18.2), (19.2), (22)

(11.3) $$\left(M_1 + d_i\right)\cdot(2 - X_{ij} - Y_{jk}) + Q_k \geq C_j + d_i, \quad i = 1,...,m\ , \quad j = 1,...,n\ , \quad k = 1,...,n$$

In addition to the constraints above, we have the valid inequality given in Section 4.3.2.1.

4.3.4. Heuristic Method for the General Problem

In the previous sections of this chapter, we presented heuristics developed for the special cases of this problem that are observed in common practical applications. All these heuristics can be used to obtain a solution for this main problem. However, we develop one more heuristic method, Heuristic 7, specially designed for this general problem.

In Theorem 1(iv), it is stated that the truck leaves a plant either as soon as it arrives there or waits for the completion of a job. The question here is, if the truck waits at the plant, how much it should wait. In other words, assuming A jobs have already been produced until the truck arrives at a plant, for how many more job completions should the truck wait. Although we could not prove it analytically, we believe that in most cases whenever the truck arrives at a plant, it follows one of the three following options:

1. Leaves the plant immediately
2. Waits for one more job
3. Waits until the truck is fully loaded

The second option means that truck will wait for the completion of the job that has already started its processing, but not completed, at the time the truck arrives to the plant.

We study the third option in Section 4.2. Although we study the first option in Section 4.3.2, since first and second options are closely related to each other, we develop Heuristic 7 to address these two options.

There are three main concepts used in this method. The first concept is the batch sliding mechanism, explained before. The second concept is the gain-loss justification, introduced in the problem with a no wait constraint. The third concept is named as *ratio-to-wait*. This concept can best be explained on an example. Suppose that the processing of job j has already started but not completed until the arrival of the truck.. If the percentage of job j's completed part is greater than or equal to *ratio-to-wait*, then the truck waits for the completion of job j, otherwise it leaves the plant immediately. Here, it is obvious that, when *ratio-to-wait* is equal to 1, the truck never waits for a job, i.e. the problem with no wait constraint. Our preliminary tests indicate that there exists no common *ratio-to-wait* value that performs well for all parameter settings. Therefore, in Heuristic 7, we let *ratio-to-wait* be equal to r, where $r = 0.1, 0.2, ..., 1$, and among these r values, choose the one yielding the smallest total completion time.

Following the discussions above, Heuristic 7 can be summarized as follows:

Step 0) Index jobs in non-decreasing order of processing times and plants in non-decreasing order of distances to the customer area. Set *totCompTime* to 0, $numSent = 0$. Calculate $\overline{p}$. Set *ratio-to-wait* = 0.1. For each plant i, set $numJobs[i] = 0$, $prodNum = 0$. Set *bestCompTime* to a very large value.

Step 1) Set $i = 1$. Determine the indexes and calculate the number of jobs that can be completely produced in the available processing time of plant i. Set $prodNum$ to this value. Decrease available processing time by the sum of processing times of these jobs. If within updated available processing time, *ratio-to-wait* of the next job can be completed, increment $prodNum$. If $prodNum \geq K$, set $numJobs[i] = K$ and go to step 2. Otherwise, go to step 4.

Step 2) Apply *batch sliding* with batch size $numJobs[i]$. Assign the resulting batch to plant i. Increase *numSent* by $numJobs[i]$. Update *totCompTime*. If $numSent < n$, go to step 1. Otherwise, update *totCompTime*, if necessary update *bestCompTime*, and go to step 3.

Step 3) If *ratio-to-wait* < 1, reset everything except *bestCompTime*, increase *ratio-to-wait* by 0.1 and go to step 1. Otherwise, return *bestCompTime* and stop.

Step 4) Increment i and calculate the number of jobs that can be produced in plant i as explained in step 1. If value found is greater than or equal to K, set $numJobs[i] = K$ and go to step 5. Otherwise, go to step 5 without changing the value of $numJobs[i]$.

Step 5) If $numJobs[i]$ satisfies *gain-loss justification* conditions, go to step 2. Otherwise, go to step 6.

Step 6) If $i < m$, go to step 4. Otherwise, set $i = 1$ and go to step 2.

A detailed pseudocode of this algorithm can be found in Appendix B.7. This heuristic runs in $O(mn^3)$ time.

4.3.5. Lower Bound for the General Problem (Lower Bound 6)

In this problem, we do not have any constraint that can let us obtain a tighter lower bound as in the previous problems. Therefore, for this problem, we calculate the lower bound analytically.

It is supposed that each plant has an equal distance $d = \min_i d_i$ to the customer. In any of the trips, it is not known whether the truck leaves the plant immediately or waits for the completion of a job. In order to find a valid lower bound, we need to suppose it leaves the

plant immediately. The only exception to this assumption is the first trip. In the first trip, the truck takes at least one job, i.e. the job with the smallest processing time (J_1), and leaves the plant either at the process completion time of J_1 or as soon as it arrives at the plant. Since the route of the truck is not known, the number of jobs to be produced until the arrival time of the truck in the following trips is not known. However, to make the lower bound valid, we assume in each trip, the truck finds K jobs already produced.

Then, with the assumptions above, the total completion time can be expressed with the following formulation:

$$K \cdot [A+d] + K \cdot [A+3d] + ... + K \cdot [A+(2q-1)d] + (n-qK) \cdot [A+(2q+1)d]$$

$$= n \cdot A + (n-qK) \cdot (2q+1) \cdot d + K \cdot d \cdot q^2$$

where $A = \max\{p_1, d\}$ and $q = \left\lfloor \frac{n}{K} \right\rfloor$.

Chapter 5

EXPERIMENTAL DESIGN AND NUMERICAL RESULTS

In this chapter, we perform experimentation and give numerical results for the methods presented in Chapter 4. In Section 5.1, the parameter settings of the test problems are presented. The results obtained by applying our methods on these test problems are given in Section 5.2.

5.1. Experimental Design

We consider five parameters and each combination of these parameters defines a different problem. These parameters are: number of plants, m; number of jobs, n; the capacity of the truck, K; processing times of the jobs, p_j, where $j = 1, 2, ..., n$; and the distances of the plants to the customer, d_i, where $i = 1, 2, ..., m$. Although our problems are difficult to solve even with two plants, we wanted to observe their performance in under a more general setting. Therefore, the number of plants, m, is set to 3.

In the choices of parameters n and K, two possible cases related to the physical sizes of the products are considered. In the first case, the company may be producing products with large physical volumes like household appliances. In the second case, the company produces products with small physical sizes. In such a case, the products are batched together

in the form of a unitized load, such as a unitized load or a box, before transported. To be able to cover both cases, a job is either a large volume product or a unitized load of small volume products. Following these discussions, the values set for n and K are given in Table 5.1. For the soft drink manufacturer mentioned in Chapter 1, 1512 products constitute a palette. In such a case, 55 jobs correspond to 83160 products and this number is usually far more than the customer demands. Hence, having at most 55 jobs is sufficient to cover even the situations with palettes with fewer products.

Table 5.1: Parameter settings for number of jobs and truck capacity

Parameters	Values
n	10, 25, 40, 55
K	2, 4, 6, 8

It is not reasonable to set the processing times of the jobs and distances of the plants to a single value as done in n and K. Instead, ranges for them, i.e. $[p_{\min}, p_{\max}]$ and $[d_{\min}, d_{\max}]$, should be determined. While determining these ranges, it was taken into consideration that a job is either a large volume product or a unitized load.

The range (D_{range}) and mean (D_{mean}) of plant distances are two main factors that may affect the performance of the proposed methods. D_{range} is defined as $(d_{\max} - d_{\min})$ while D_{mean} is defined as $(d_{\max} + d_{\min})/2$. In order to be able to determine the effect of these factors, plant distances used in the test problems are generated from the following three ranges:

- Plant Distance Range 1 : $D_{range} = 6$, $D_{mean} = 25$
- Plant Distance Range 2 : $D_{range} = 6$, $D_{mean} = 35$

- Plant Distance Range 3 : $D_{range} = 26$, $D_{mean} = 35$

To determine the effect of range (P_{range}) of processing times on the performance of the proposed methods, processing times are first grouped into two as small and large ranges. For the assessment of the effect of mean (P_{mean}), small range group is divided into three subgroups as small, medium and large mean. Processing times used in the test problems are generated from the following four ranges:

- Processing Time Range 1 : $P_{range} = 10$, $P_{mean} = 6$
- Processing Time Range 2 : $P_{range} = 10$, $P_{mean} = 10$
- Processing Time Range 3 : $P_{range} = 10$, $P_{mean} = 30$
- Processing Time Range 4 : $P_{range} = 34$, $P_{mean} = 18$

Hence, as seen in Table 5.2, there are twelve ([$p_{\min}$, $p_{\max}$], [$d_{\min}$, $d_{\max}$]) combinations from which the processing times and plant distances are generated by using a uniform distribution. For each of these 12 combinations, there exist 16 different (n , K) pairs. Therefore, a total of 192 test problems are created and for each of these problems 5 instances are generated randomly.

All heuristics and lower bound algorithms are coded in C++ and run on a PC with a 1.8 GHz AMD Turion 64 processor and 512 MB memory (recall that these procedures run in polynomial time). The exact methods are solved by calling the MIP solver of GAMS on a computer with 3.73 GHz, 2 Dual-Core Intel Hyperthreading Xeon CPU processor and 24 GB shared memory.

Table 5.2 : Parameter settings for processing times and plant distances

Processing Time Range 1	$p_j \in [1,11],\ d_i \in [22,28]$	**Processing Time Range 3**	$p_j \in [25,35],\ d_i \in [22,28]$
	$p_j \in [1,11],\ d_i \in [32,38]$		$p_j \in [25,35],\ d_i \in [32,38]$
	$p_j \in [1,11],\ d_i \in [22,48]$		$p_j \in [25,35],\ d_i \in [22,48]$
Processing Time Range 2	$p_j \in [5,15],\ d_i \in [22,28]$	**Processing Time Range 4**	$p_j \in [1,35],\ d_i \in [22,28]$
	$p_j \in [5,15],\ d_i \in [32,38]$		$p_j \in [1,35],\ d_i \in [32,38]$
	$p_j \in [5,15],\ d_i \in [22,48]$		$p_j \in [1,35],\ d_i \in [22,48]$

5.2. Numerical Results

In this section, the results of the proposed methods for the problems studied are given in the order they are presented in Chapter 4. For each problem, the results are grouped according to their processing time ranges. The impacts of the number of jobs, truck capacity, and plant distance range will be given under these groups.

To assess the performance of the proposed heuristics, firstly the percentage gaps between the total completion times obtained by the heuristics and the corresponding lower bounds, named as *LB gap*, are calculated by using the following formula:

$$LB\ Gap = \frac{\left(Heuristic\ Value - Lower\ Bound\ Value\right)}{Lower\ Bound\ Value} \cdot 100$$

If the calculated gaps are large, optimal solutions are found for instances with small number of jobs. By comparing the heuristic results of these instances with the optimal results, we

attempted to determine the cause of the large heuristic-lower bound gap: poor performance of the heuristic or loose lower bound. The percentage gaps between heuristic results and optimal results, named as *optimality gap*, are calculated by the following formula:

$$Optimality\ Gap = \frac{\left(Heuristic\ Value - Optimal\ Value\right)}{Optimal\ Value} \cdot 100$$

In the tables in Appendix C, D, E and F, average and maximum *LB gap* and *optimality gap* values are also given. These values are calculated by taking the average or maximum of the results of 5 random instances for each problem.

5.2.1. Numerical Results and Discussions for Last Trip Partial Problem

As explained in Chapter 4, one exact algorithm, MIP-1, two heuristic algorithms, Heuristics 1 and 2, and one lower bound generation algorithm are developed for this problem. The results obtained by the proposed methods are given in Appendix C.

Processing Time Range 1:

In this range, both heuristics yield the same or very close total completion times and these values are optimal at least for 67 of 80 tests. For the rest, the gap between the heuristic results and lower bound is very small (see Tables C.1, C.2 and C.3 in Appendix C for details). For all plant distance ranges in this group, *LB gap* values are 0% when there are 10 jobs to be produced and distributed. For other number of jobs values ($n = 25, 40, 55$), *LB gaps* are shown in Figures 5.1 and 5.2 for Heuristic 1 and Heuristic 2, respectively.

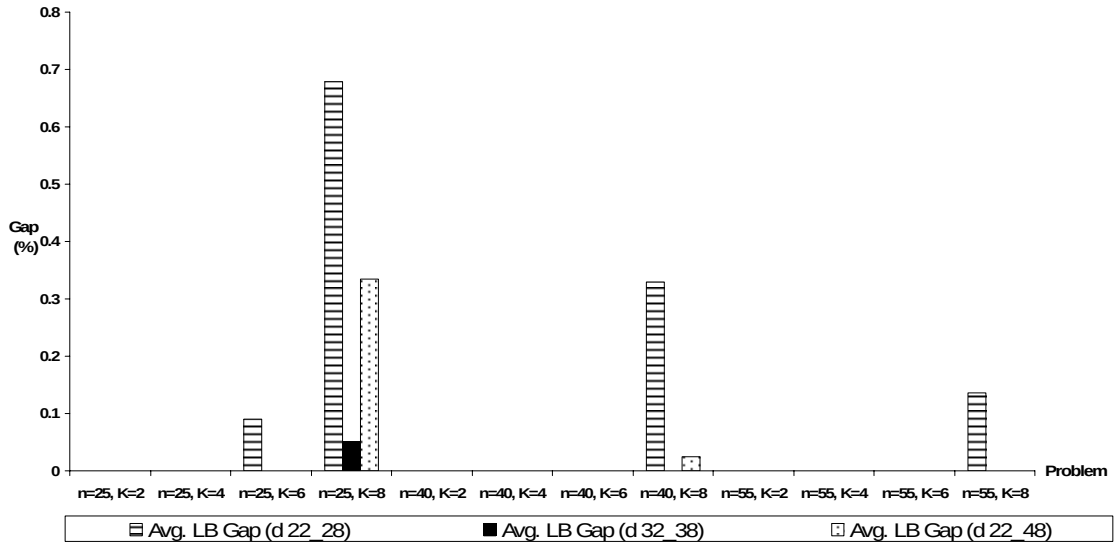

Figure 5.1: Numerical results for Heuristic 1 for $n = 25, 40, 55$ where $p_j \in [1, 11]$

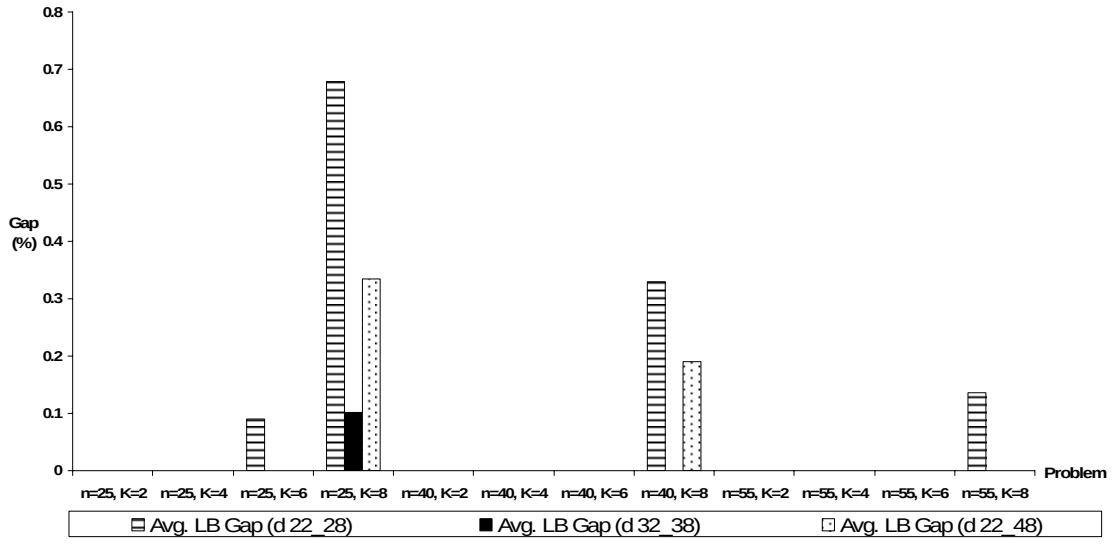

Figure 5.2: Numerical results for Heuristic 2 for $n = 25, 40, 55$ where $p_j \in [1, 11]$

By comparing Figures 5.1 and 5.2 it can be seen that Heuristic 1 performs better than Heuristic 2 in this processing time range. Although there are six data to be shown in the plots, for some (n, K) pairs, there are less than six bars. This means that *LB gap* was zero for

these parameter values. Since the mean of the processing times is small when compared to the mean of plant distances, the truck can make fully loaded trips only by visiting closer plants. As the capacity of the truck increases, the first trip becomes problematic because there may not be K jobs produced until the truck arrives to the first plant visited. As the mean of the plant distances increases, the probability that K jobs would be produced until the arrival of the truck to the first visited plant increases. Hence, *LB gap* decreases. This can be observed by comparing the 1st and 3rd bar columns for each (n, K) pair, in Figures 5.1 and 5.2. The impact of a change in the range of plant distances can be seen by comparing the 3rd and 5th bar columns for each (n, K) pair, in Figures 5.1 and 5.2. As the range increases, although the mean remains the same, the distance of the closest plant to the customer decreases. This decreases the probability of producing K jobs until the arrival of the truck on the first trip. Hence, *LB gap* increases.

Processing Time Range 2:

This range has larger processing times than the previous one. Because of this increase in processing times, it is not possible to send the truck fully loaded by always visiting the closest plant. Especially, when the truck capacity is large, i.e. $K = 6$ or 8, in the proposed heuristics the truck may have to visit the second closest plant on the second or a later trip. Since the lower bound is calculated by considering that each plant has the same distance to the customer, i.e. the distance of the closest plant, the usage of a farther plant causes *LB gap* to be greater than zero. For 10 jobs, the *LB* and *optimality gaps* are shown in Figures 5.3 and 5.4 for Heuristic 1 and Heuristic 2, respectively. As can be seen in these figures, Heuristic 1 yields optimal results for this processing time range. For more than 10 jobs, since obtaining the optimal result takes a long time, exact algorithm is used only for the instances with *LB gap* larger than 10%. Maximum *LB gap* is less than 6% and hence only the *LB gaps* are given for problems with more than 10 jobs. Detailed test results for this processing time range can be found in Tables C.4, C.5 and C.6 in Appendix C.

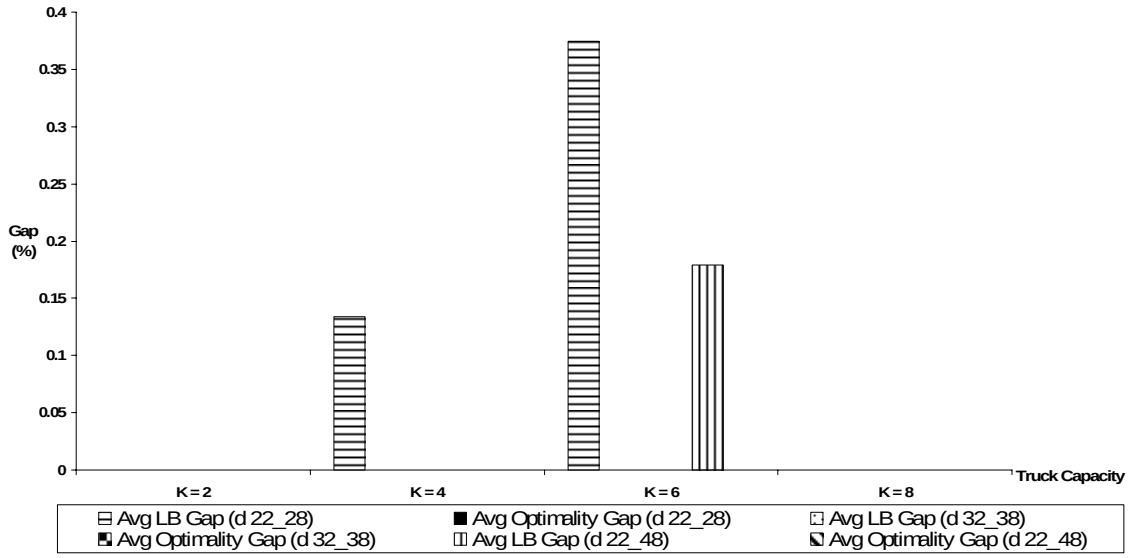

Figure 5.3: Numerical results for Heuristic 1 for $n = 10$ where $p_j \in [5,15]$

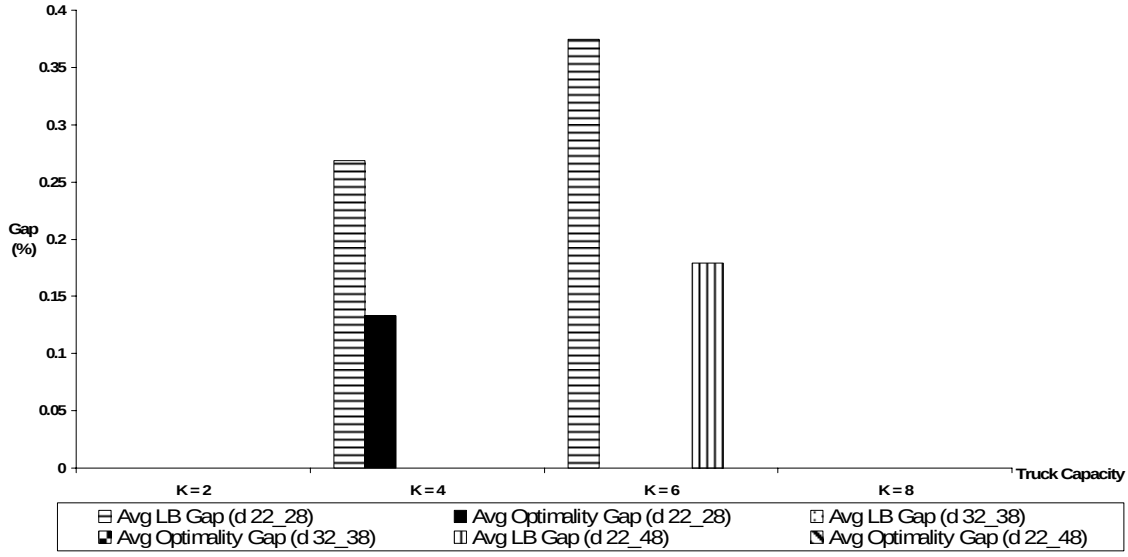

Figure 5.4: Numerical results for Heuristic 2 for $n = 10$ where $p_j \in [5,15]$

As seen by comparing 1st and 3rd bar columns for each (n, K) pair in Figures 5.5 and 5.6, an increase in the mean of the plant distance range decreases *LB gap* because of the

reasons explained in Processing Time Range 1. As the range of the plant distances increases, the distance difference between the closest plant and the second closest plant increases. Hence, when the second closest plant is visited, the deviation from the lower bound will be larger for the instances that have larger difference between the closest and farthermost plant, i.e. larger plant distance range.

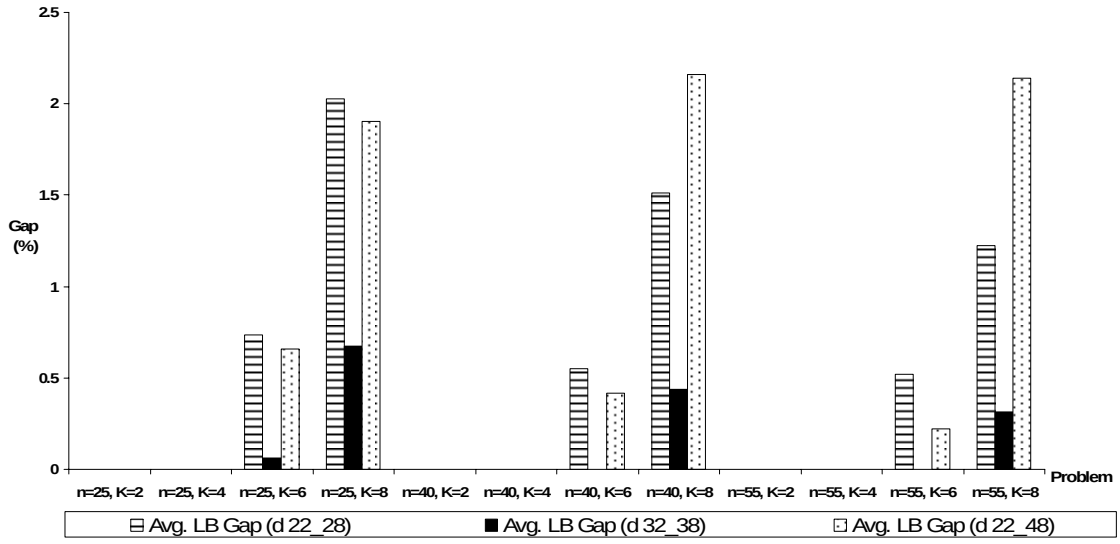

Figure 5.5: Numerical results for Heuristic 1 for $n = 25, 40, 55$ where $p_j \in [5,15]$

It should be noted that, for this processing time range, Heuristic 2 outperforms Heuristic 1 in most of the instances. The reason for this performance difference is inherent in the ways these heuristics make the choices of jobs to assign to a selected plant. When the truck visits the second closest plant, Heuristic 2 assigns the K-size batch with greatest processing time to this plant, and is left with the jobs having smaller processing times. Hence, the truck would be able to transport K jobs from the closest plant on its next trip. However, since Heuristic 1 assigns the first unassigned K jobs to the second closest plant, it is left with jobs having larger processing times. Then the truck cannot transport K jobs from the closest plant in its next trip and hence needs to make the trip to a farther plant and this causes a delay in trip completion times.

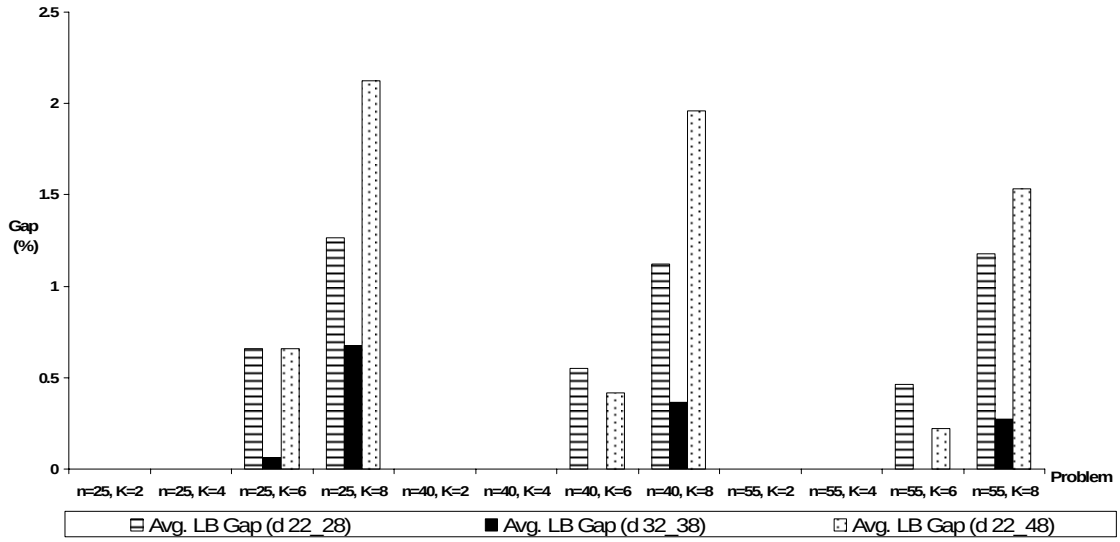

Figure 5.6: Numerical results for Heuristic 2 for $n = 25, 40, 55$ where $p_j \in [5,15]$

Processing Time Range 3:

The processing times used in this range are so large that even if the truck capacity is set to the smallest value, i.e. $K = 2$, it is not possible to fill the truck by visiting the closest plant in every trip. Therefore, the truck requires visiting farther plants and, as explained above, this causes a deviation from the lower bound yielding *LB gaps* larger than 0. In Figures 5.7 and 5.8, it can be seen that *LB gaps* are very small for 10 jobs. For more than 10 jobs, all average *LB gaps* are smaller than 10% as can be seen in Figures 5.9 and 5.10. The test results for this range are given in Tables C.7, C.8 and C.9 in Appendix C.

Here, one important point to mention is that the *LB gaps* in this range are higher than the gaps of previous ranges. The reason for this is the increase in the processing times. As processing times increase, the probability that the truck finds K jobs already produced whenever it arrives to a plant decreases. However, the lower bound is calculated based on the assumption that the truck finds K jobs. Therefore, lower bound is not tight in this range as it is in the previous ranges.

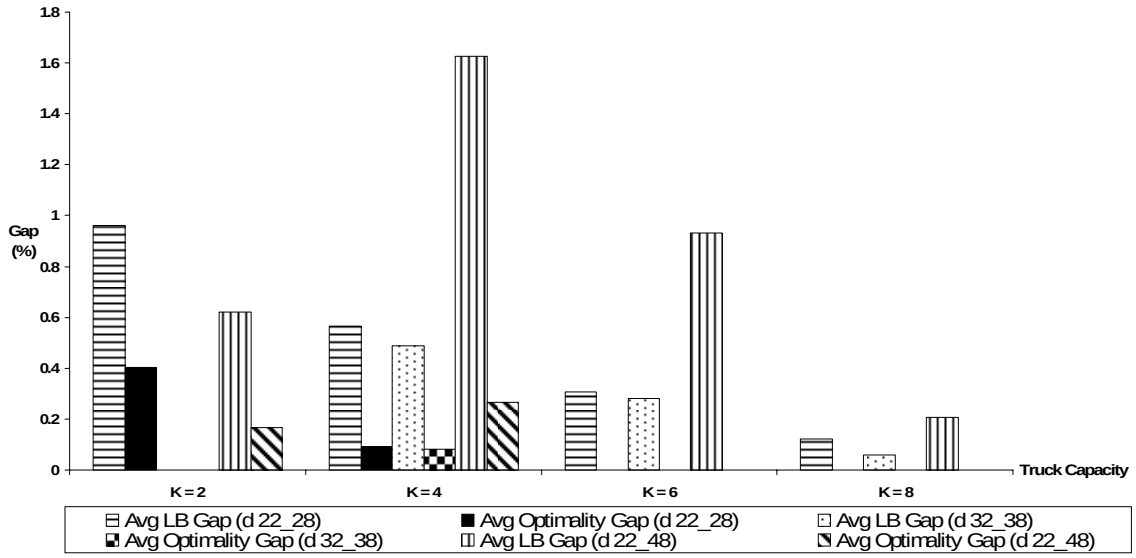

Figure 5.7: Numerical results for Heuristic 1 for $n = 10$ where $p_j \in [25, 35]$

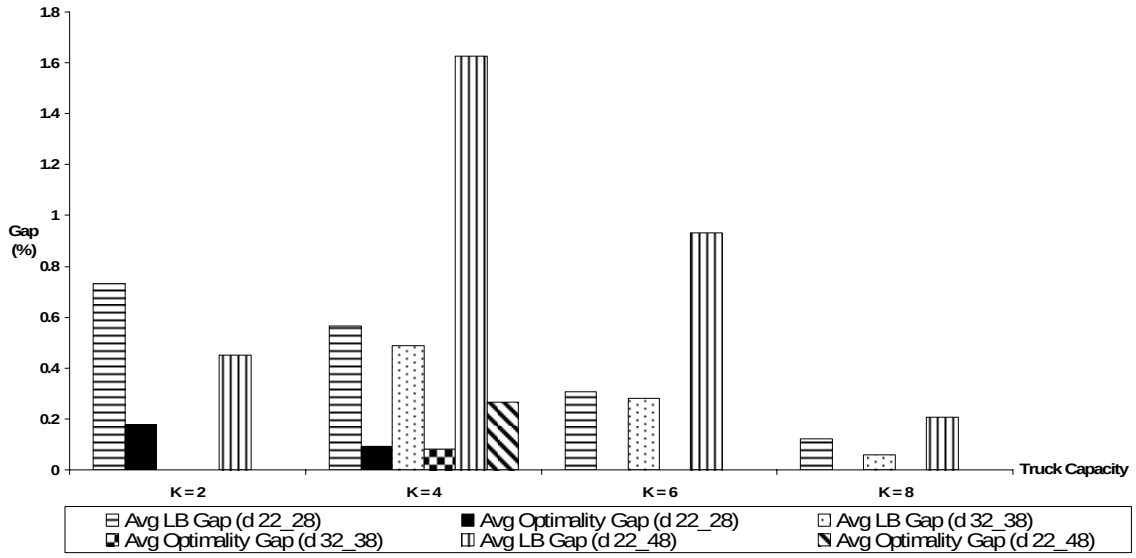

Figure 5.8: Numerical results for Heuristic 2 for $n = 10$ where $p_j \in [25, 35]$

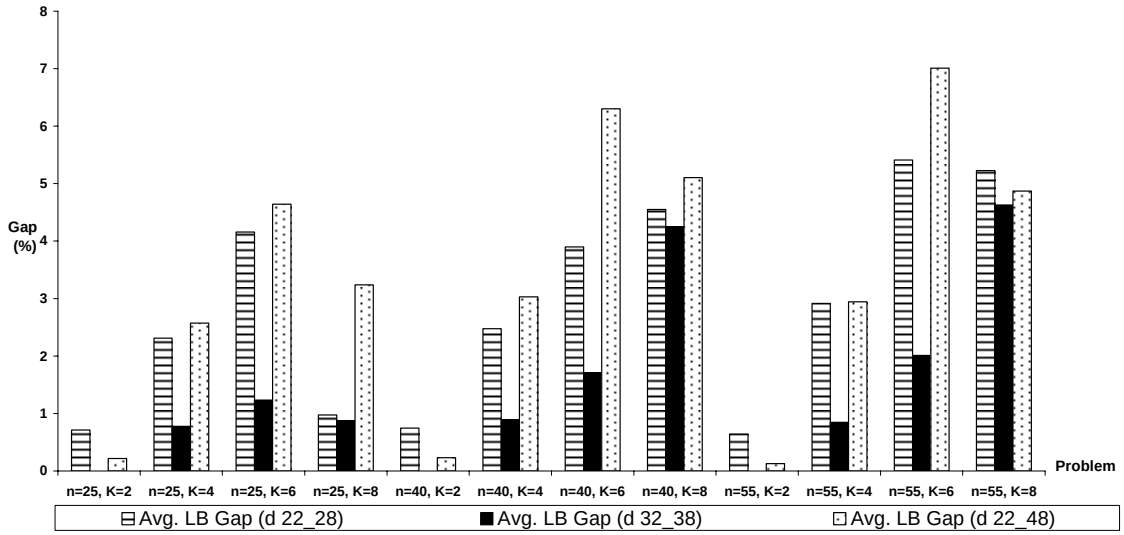

Figure 5.9: Numerical results for Heuristic 1 for $n = 25, 40, 55$ where $p_j \in [25, 35]$

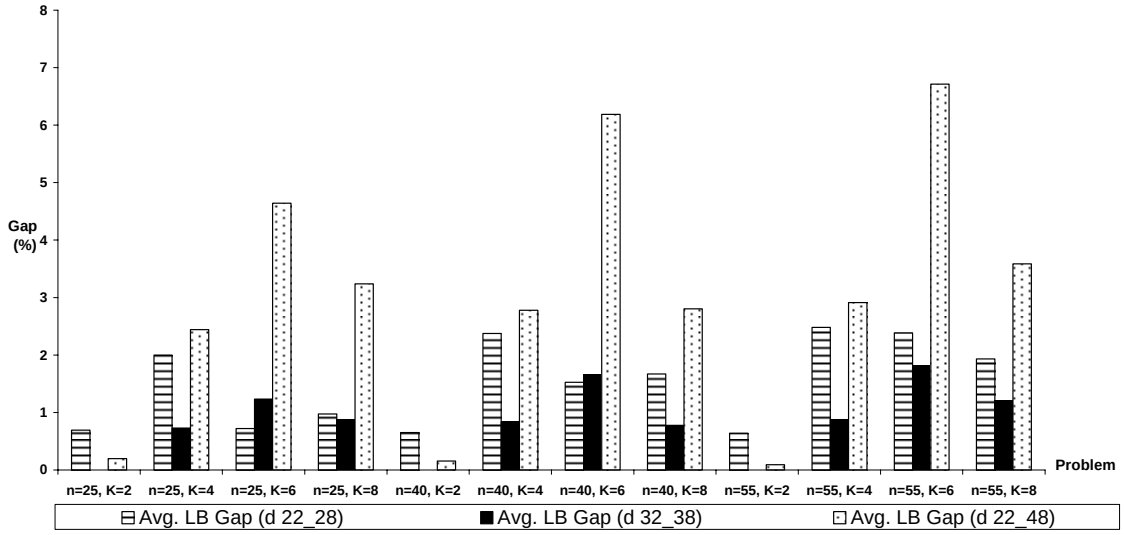

Figure 5.10: Numerical results for Heuristic 2 for $n = 25, 40, 55$ where $p_j \in [25, 35]$

In general, as explained for the previous ranges, an increase in the mean of the plant distances causes a decrease in average *LB gap*. However, in Figure 5.10 for $(n = 25, K = 6)$ and $(n = 40, K = 6)$, it is observed that an increase in the mean plant distance causes an increase in average *LB gap*. The reason for this unexpected situation can be explained as follows. In these problems, since the processing times are large, farther plants are also visited. When the closest plant is visited after visiting farther plants, for the problems with $d_i \in [32, 38]$, the truck finds K jobs already produced and the updating mechanism in the lower bound generation does not work. However, for the problems with $d_i \in [22, 28]$, the truck does not find K jobs produced as it arrives to the closest plant and the updating mechanism in the lower bound works and hence *LB gap* is smaller.

Processing Time Range 4:

This is the processing time range that has the largest *LB gap* values. Since the range of the processing times is large, updating mechanism in the lower bound generation method may not work and hence obtained lower bounds may be looser than the previous ranges. As can be seen in Figures 5.11 and 5.12, *LB gaps* around 20% exist for 10 jobs. However, the maximum *optimality gap* is smaller than 2.5%. In Figures 5.13 and 5.14, for the problem with $(n = 25, K = 8)$, high *LB gap* values are observed. For these instances with *LB gap* value greater than 10%, exact methods could not yield the optimal solution in a short time. By increasing the resource limit to 50000 seconds, we obtained the optimal solution except for 3 of the instances (see Tables C.10, C.11 and C.12 in Appendix C for details). As seen in Tables 5.3 and 5.4, proposed heuristics yield close to optimal results.

Table 5.3 : *LB* and *optimality gaps* for Heuristics 1 and 2 for $(n = 25, K = 8)$ and $d_i \in [22, 28]$

n	K	LB Gap (%)		Optimality Gap (%)	
		Heuristic 1	**Heuristic 2**	**Heuristic 1**	**Heuristic 2**
25	**8**	39.98	39.98	1.10	1.10
		29.67	29.67	0.76	0.76
		41.89	41.89	1.58	1.58
		17.94	17.94	0.77	0.77
		17.72	17.72	0.35	0.35

Table 5.4 : *LB* and *optimality gaps* for Heuristics 1 and 2 for $(n = 25, K = 8)$ and $d_i \in [22, 48]$

n	K	LB Gap (%)		Optimality Gap (%)	
		Heuristic 1	**Heuristic 2**	**Heuristic 1**	**Heuristic 2**
25	**8**	12.64	12.64	0.40	0.40
		20.59	20.59	0.73	0.73
		33.31	33.84	1.40	1.80

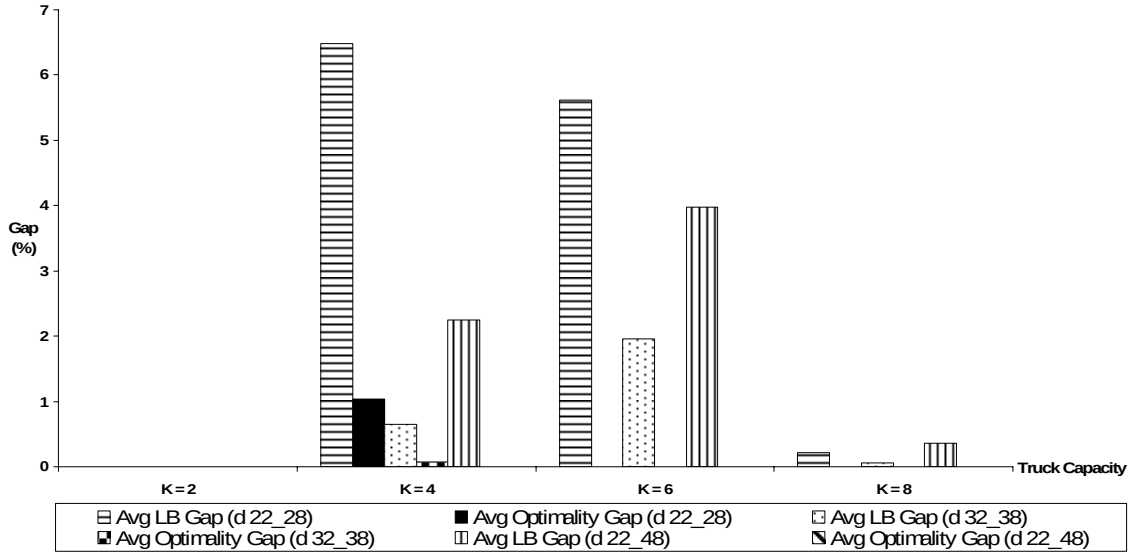

Figure 5.11: Numerical results for Heuristic 1 for $n = 10$ where $p_j \in [1, 35]$

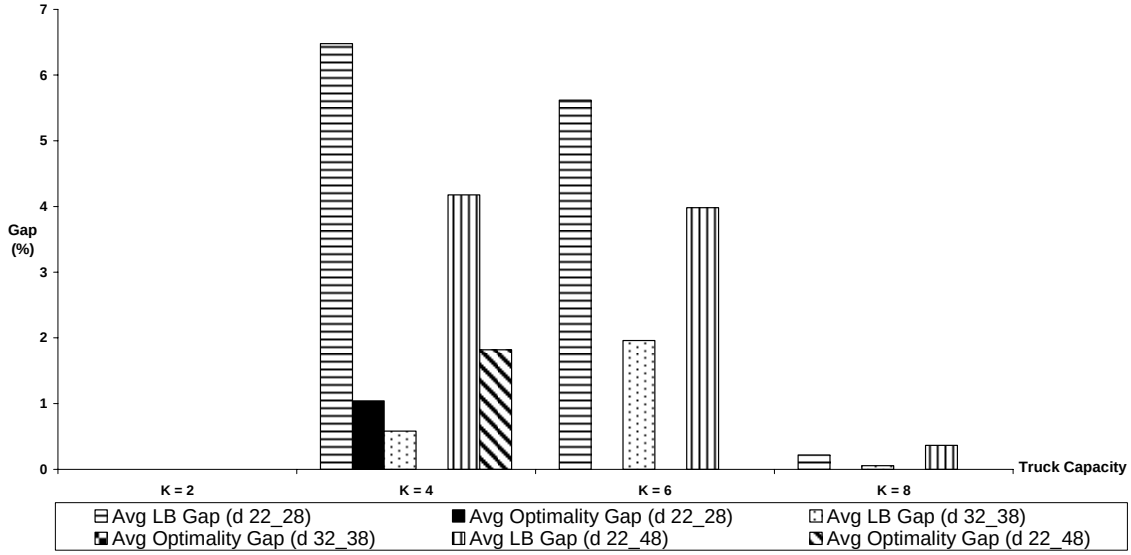

Figure 5.12: Numerical results for Heuristic 2 for $n = 10$ where $p_j \in [1,35]$

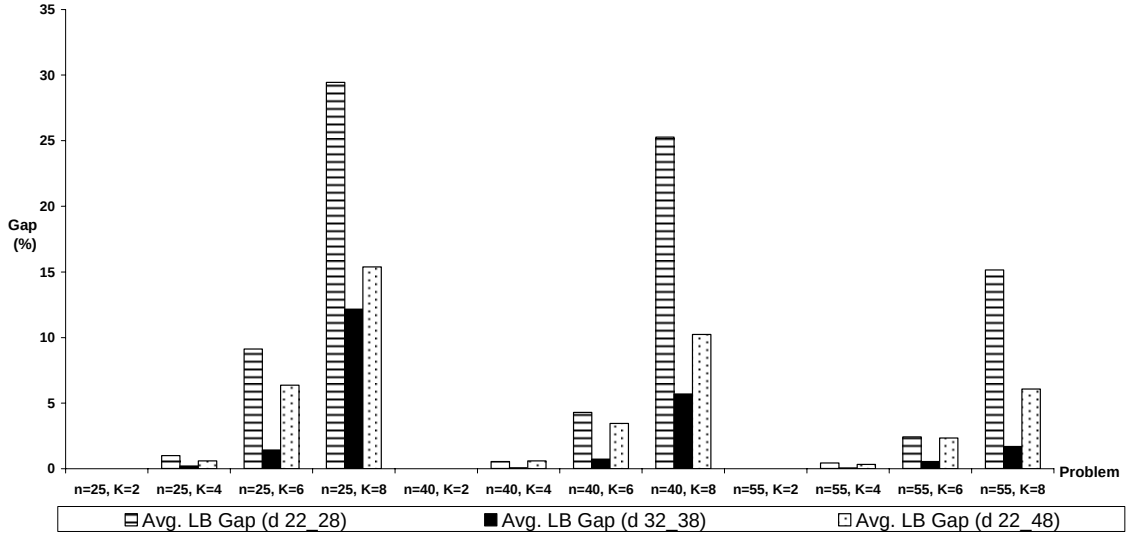

Figure 5.13: Numerical results for Heuristic 1 for $n = 25, 40, 55$ where $p_j \in [1,35]$

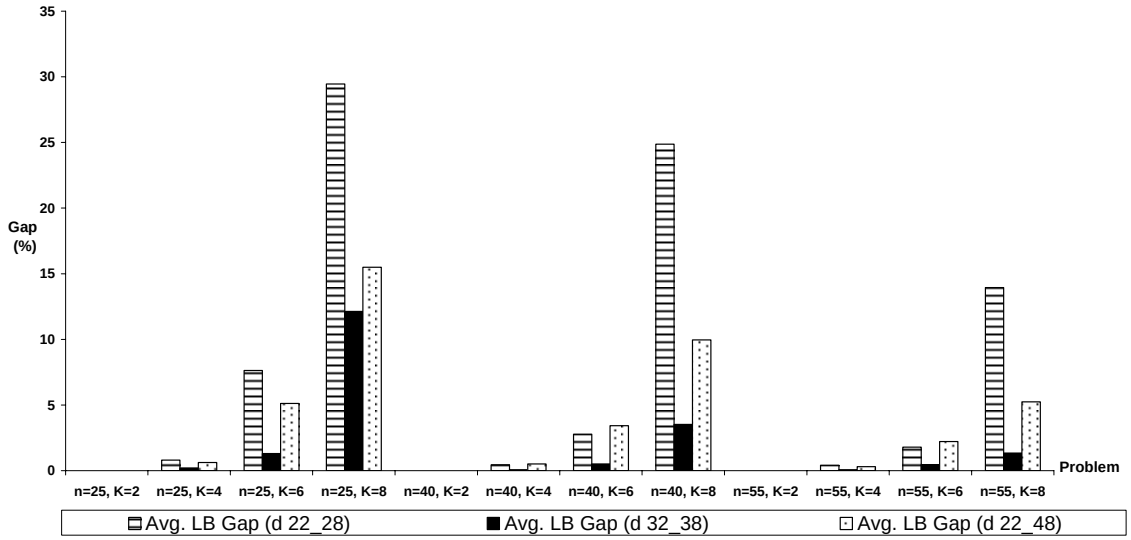

Figure 5.14: Numerical results for Heuristic 2 for $n = 25, 40, 55$ where $p_j \in [1, 35]$

As in the previous processing time ranges, in this range, average *LB gap* decreases as the mean of the plant distance range increases. An increase in the range, increases average *LB gap*.

5.2.2. Numerical Results and Discussions for Fully Loaded Trips Problem

In this case, although we develop three different lower bound generation methods, in the results we use only two of them since Lower Bound 2 always yields worse results. It is not possible to say that Lower Bound 3 outperforms Lower bound 4 or the opposite. Therefore, for each instance we choose the one yielding larger lower bound value between these two methods. The lower bound values obtained and other test results for this problem is given in Appendix D.

Processing Time Range 1:

In this range, the gap between the heuristic result and the lower bound is small in general. The only problematic case is the problem with $n = 10$ and $K = 8$. For this problem, for all plant distance ranges, *LB gap* is large, however, as shown in Figure 5.15 for 10 jobs, proposed heuristics generate near optimal solutions. Here, only the figure for Heuristic 3 is given because the results of Heuristics 3 and 4 are the same for 10 jobs.

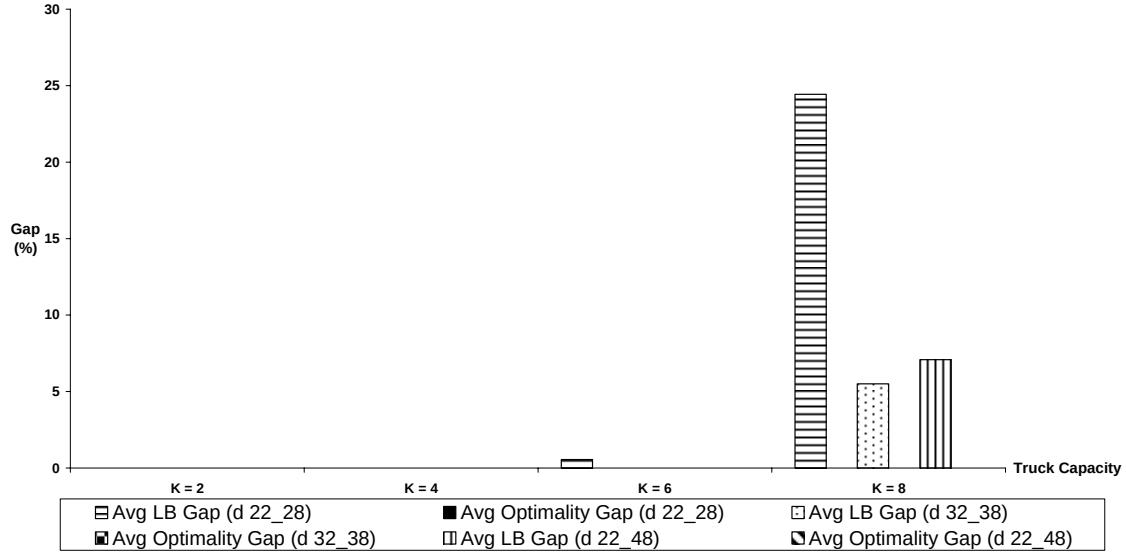

Figure 5.15: Numerical results for Heuristic 3 for $n = 10$ where $p_j \in [1,11]$

Since the heuristics can provide near optimal solutions, the reason for large *LB gaps* is the poor performance of the lower bound. The reason for this performance is inherent in the calculation of completion time of the first trip. Since it is not known which trip would be partially loaded, we suppose in the lower bound that the first trip is partially loaded and calculate the departure time of the truck from the first visited plant accordingly. However, in order to obtain a valid lower bound, we calculate the lower bound as if the truck transports K jobs in its first trip with this departure time. When the truck capacity is large, it is not possible to send the first truck fully loaded from the closest plant and this causes the lower bound to be loose. As seen in Figures 5.16 and 5.17, when more than 10 jobs require

transportation, this effect of the first trip vanishes because of the updating mechanism on the first trips of each round. This is the reason for small *LB gaps* with large number of jobs.

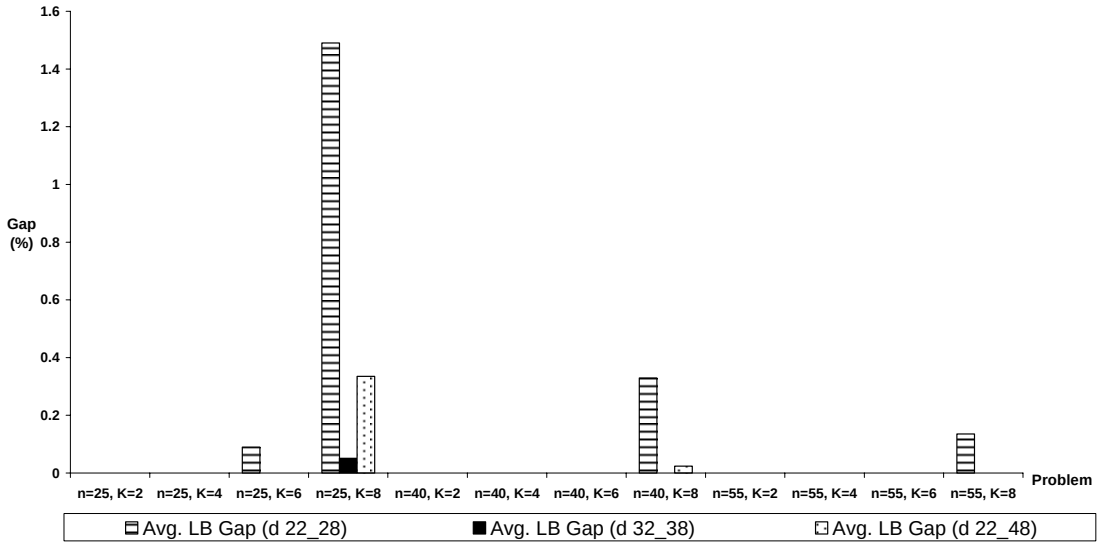

Figure 5.16: Numerical results for Heuristic 3 for $n = 25, 40, 55$ where $p_j \in [1, 11]$

As seen by comparing the 3rd and 5th column bars for each (n, K) pair in Figures 5.16 and 5.19, *LB gap* decreases as mean of the plant distance range increases. As plant distances increases, the probability that the truck leaves a plant immediately and fully loaded increases. Since this is what we suppose in the calculations of the lower bound, the decrease in the *LB* gap with the increase in distance range is in compliance with our expectations. In fact, this situation holds not only for this range but for all processing time ranges. Therefore, we can conclude that for the full truck case, as the mean of the plant distance range increases, *LB gap* decreases.

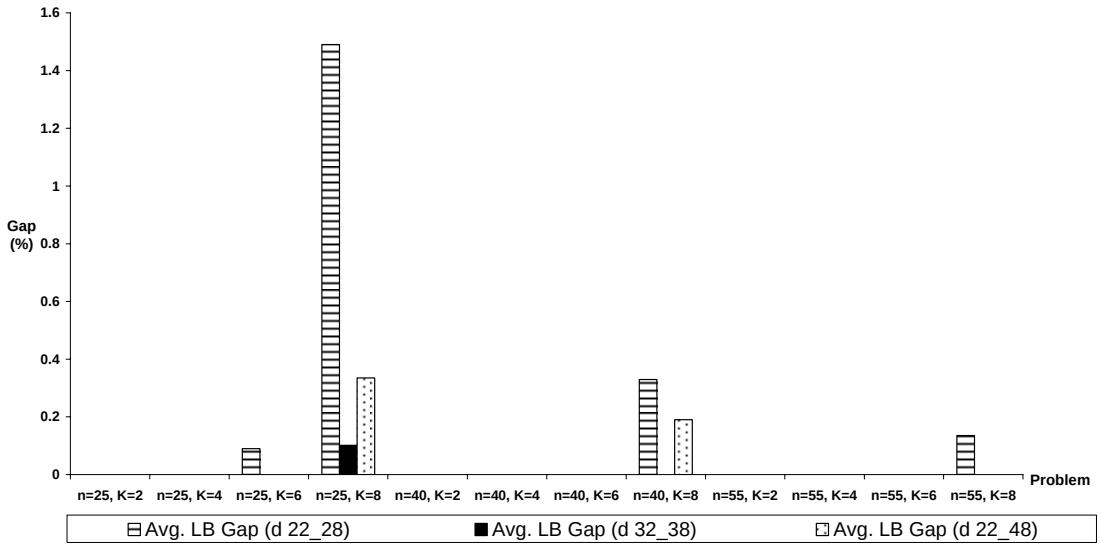

Figure 5.17: Numerical results for Heuristic 4 for $n = 25, 40, 55$ where $p_j \in [1, 11]$

Since the processing times are small and it is possible to send the truck fully loaded without causing it to wait at a plant, sending only the last truck partially loaded is better from the perspective of obtaining smaller total completion times. This result can be observed by comparing Table D.1 with C.1; Table D.2 with C.2 and Table D.3 with C.3.

Processing Time Range 2:

In this range, since the processing times are larger than range 1, sending the last truck partially loaded is better only if the capacity of the truck is small. As truck capacity increases, it gets harder to fill the truck in the first trips and hence waiting at a plant for the truck to be fully loaded in the first trips causes a delay in the trip completion times of the following trips. Also, while generating the lower bound, since it was assumed that the truck can be fully loaded as soon as it arrives to a plant, as the truck capacity increases and the probability of filling the truck in the first trips decreases, *LB gap* increases. In Figures 5.18 and 5.19, for 10 jobs, it is shown that *LB gaps* can be very large due to poor performance of the lower bounds

because of the reason mentioned above. The maximum optimality gap obtained by Heuristic 4 is 5.86%, seen in the distance range [22,28].

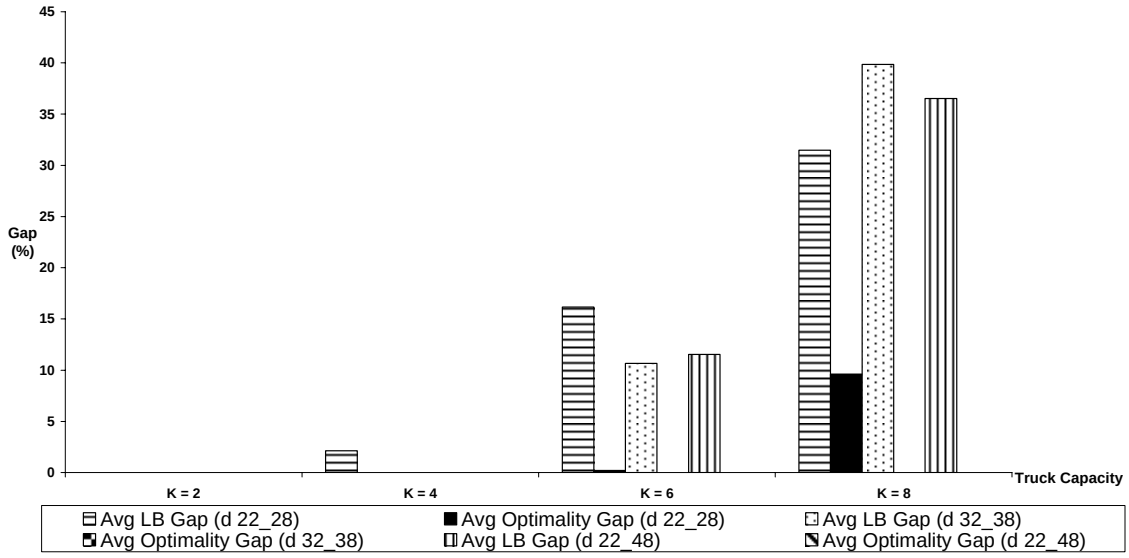

Figure 5.18: Numerical results for Heuristic 3 for $n = 10$ where $p_j \in [5,15]$

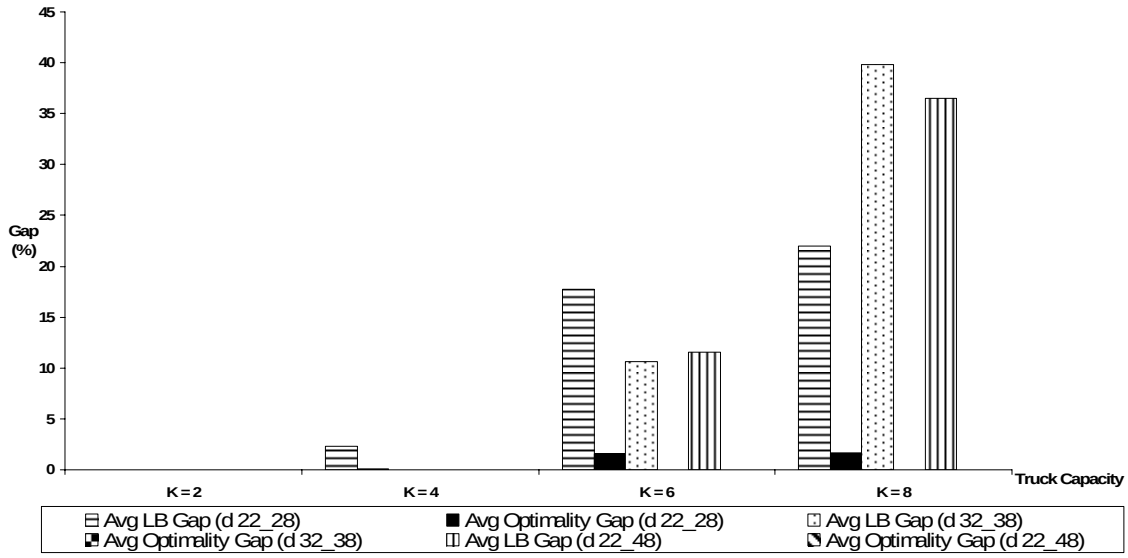

Figure 5.19: Numerical results for Heuristic 4 for $n = 10$ where $p_j \in [5,15]$

As can be seen both from Figures 5.18 and 5.19, and from Figures 5.20 and 5.21, increasing the mean of the plant distance range decreases *LB gaps*. The reason for this result was explained in Processing Time Range 1.

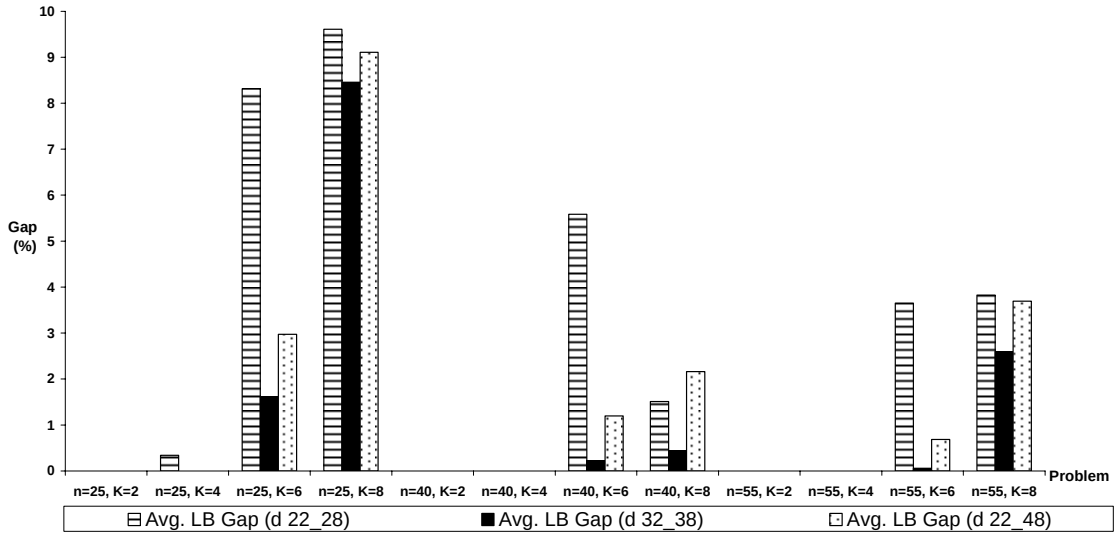

Figure 5.20: Numerical results for Heuristic 3 for $n = 25, 40, 55$ where $p_j \in [5,15]$

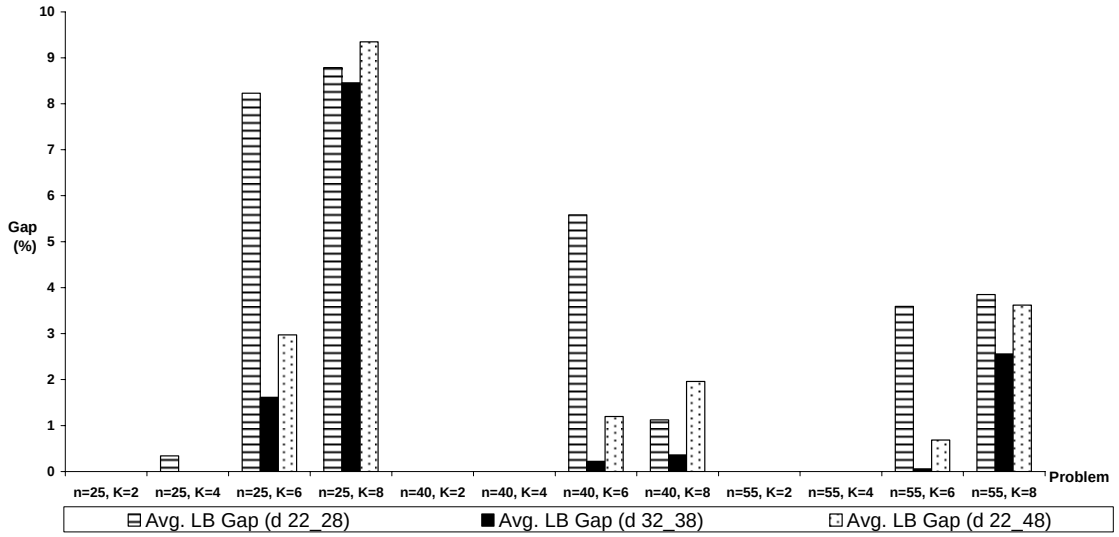

Figure 5.21: Numerical results for Heuristic 4 for $n = 25, 40, 55$ where $p_j \in [5,15]$

For 25 jobs, we solved MIP-2 for the instances that have *LB gaps* either larger or close to 10%. However, even in 50000 seconds we could not obtain the optimal solution for those problems. In Tables D.4 and D.5 in Appendix D, the best integer values found at the end of 50000 seconds are given in the optimal results column with a (*) mark.

Processing Time Range 3:

The increase in processing times in this range causes an increase in *LB gap* values. As explained in the results for the problem with only last trip partially loaded, in this processing time range, it is not possible to fill the truck in each trip as is supposed in the lower bound generation. Hence, lower bound values are much smaller when compared to the optimal results. Therefore, large *LB gaps* will be observed as seen in Figures 5.22, 5.23, 5.24 and 5.25. The results for this range are given in Tables D.4, D.5 and D.6 in Appendix D.

In Figures 5.22 and 5.23, it is shown for 10 jobs that proposed heuristics yield optimal or near-optimal results. For 25 jobs, we tried obtaining the optimal results for the instances with *LB gap* values larger than 10% but could not get the results even in 13.89 hours (50000 seconds).

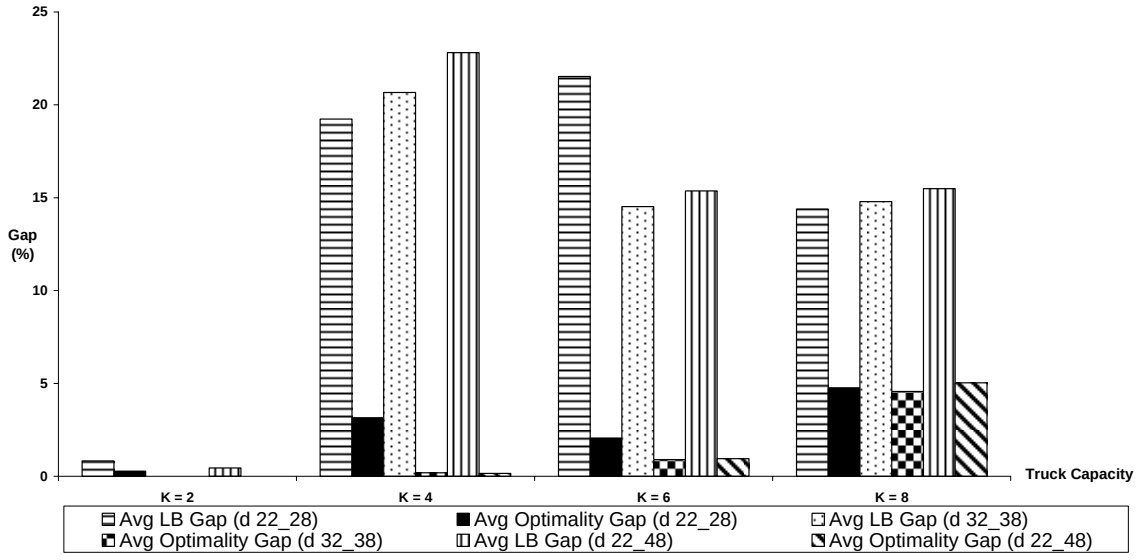

Figure 5.22: Numerical results for Heuristic 3 for $n = 10$ where $p_j \in [25,35]$

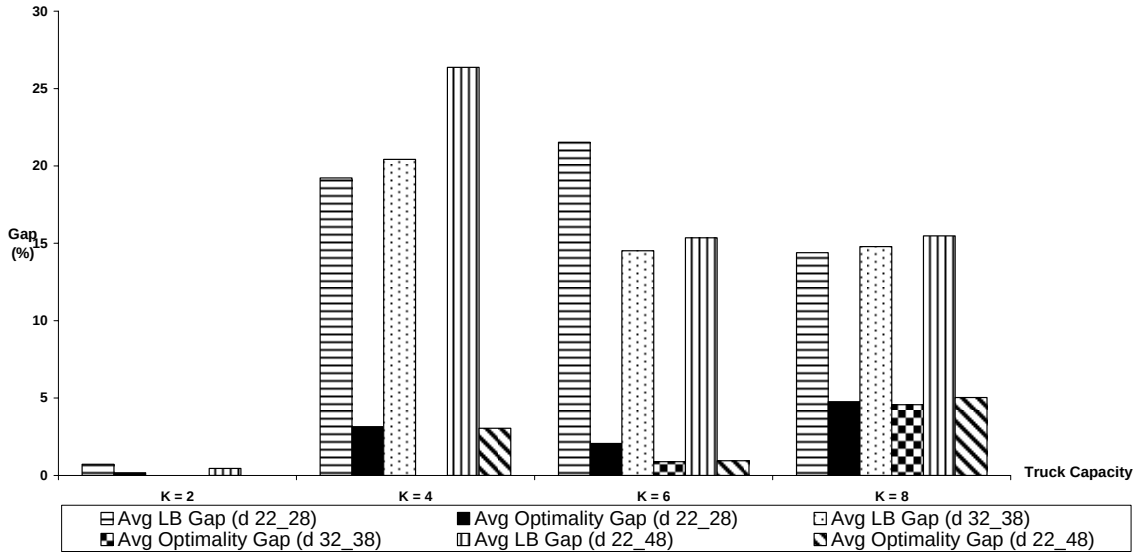

Figure 5.23: Numerical results for Heuristic 4 for $n = 10$ where $p_j \in [25,35]$

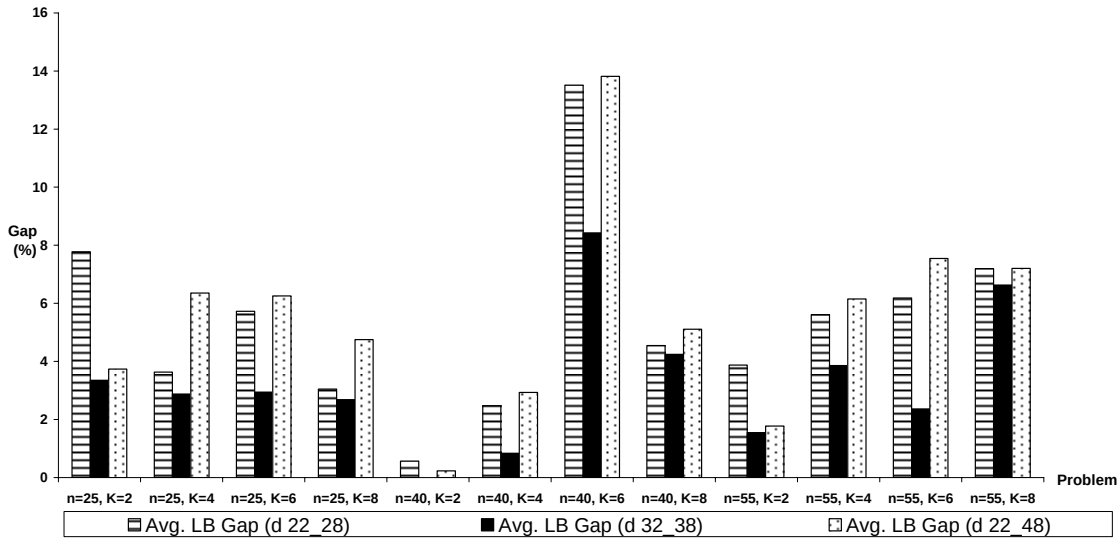

Figure 5.24: Numerical results for Heuristic 3 for $n = 25,40,55$ where $p_j \in [25,35]$

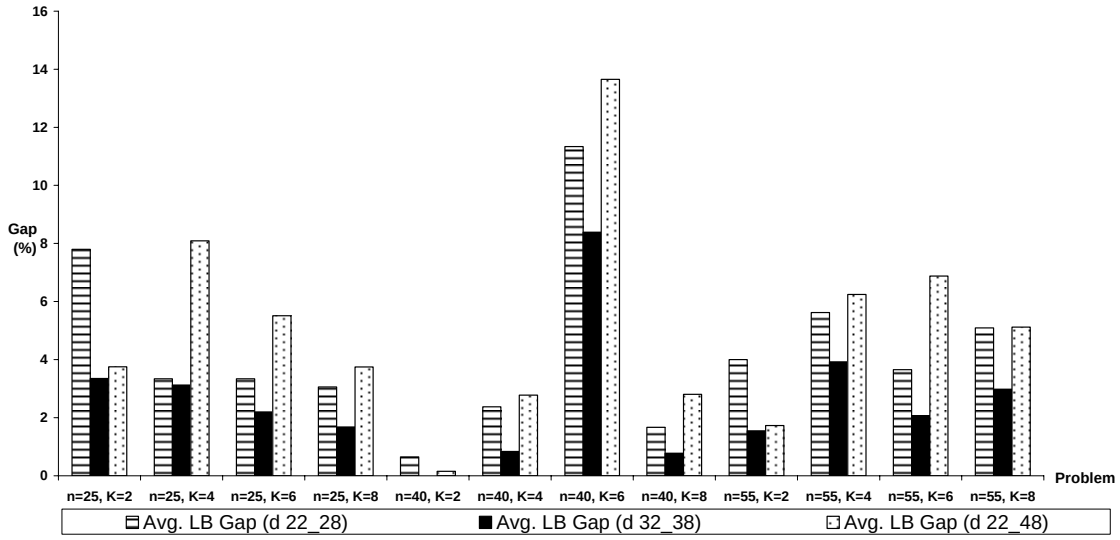

Figure 5.25: Numerical results for Heuristic 4 for $n = 25, 40, 55$ where $p_j \in [25, 35]$

Although can be observed by comparing Table D.5 with A.5; Table D.6 with C.6 and Table D.7 with C.7, it is possible to say intuitively that, sending the first trips fully loaded and the last trip partially loaded would not yield good results in terms of smaller total completion times. Since the processing times are large, for the trips in the beginning of the schedule, the truck will need to wait for the completion of a job at a plant in order to become fully loaded. And this will delay the following trips most of the time unnecessarily.

Processing Time Range 4:

In this range for 10 jobs *optimality gap* can be as high as 15.71% as seen in Figures 5.26 and 5.27. The reason for this is that in our heuristics, if the truck requires taking l jobs from a plant and it can find that amount of jobs already produced as it arrives to that plant, it leaves that plant immediately. However, with this processing time range, in the optimal solution, even if the truck finds l jobs already produced as it arrives to a plant; it waits for the completion time of l largest jobs. Hence, in the following trip, the truck can be fully loaded by either immediately leaving the plant or after waiting a small amount of time. For

example, in the second instance of the problem with $n=10$ and $K=8$, we have $p=\{3,5,10,11,19,21,31,31,34,34\}$ and $d=\{24,25,28\}$. Both of our heuristic methods choose sending the last trip partial and transports jobs with processing times $\{3,5,10,11,19,21,31,31\}$ from the closest plant on the first trip and the rest from the second plant on the second trip. However, in the optimal solution, jobs with processing times $\{34,34\}$ are assigned to the second plant but they are transported on the first trip. After transporting these jobs, the truck goes to the closest plant and waits for the completion time of the remaining eight jobs and transports them. Hence, both in our methods and in the optimal solution, the completion time of the first eight jobs are the same. But the completion time of the last two jobs in the optimal solution is much smaller than the completion time found by our proposed heuristics.

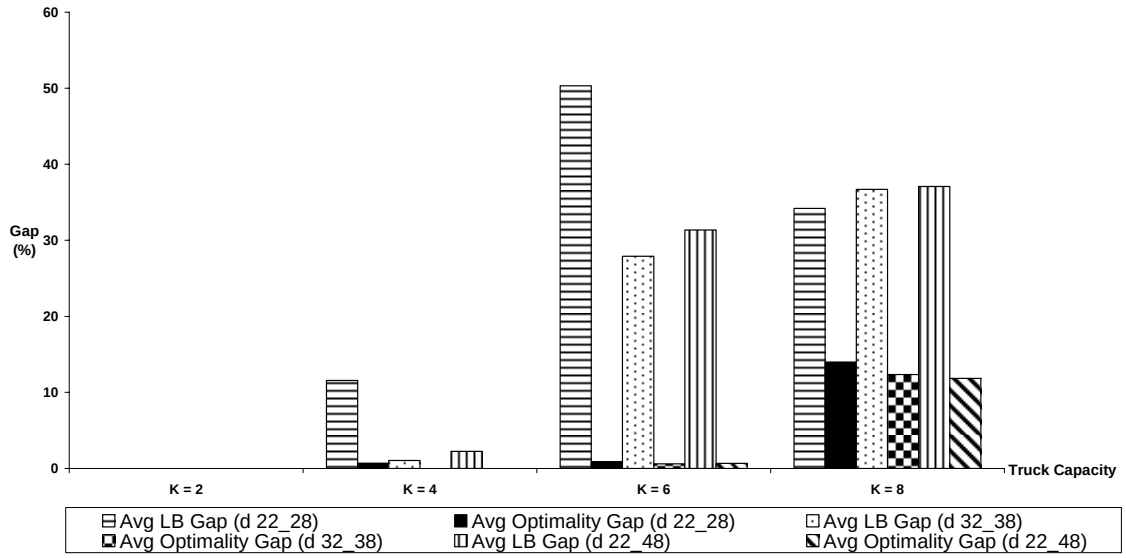

Figure 5.26: Numerical results for Heuristic 3 for $n=10$ where $p_j \in [1,35]$

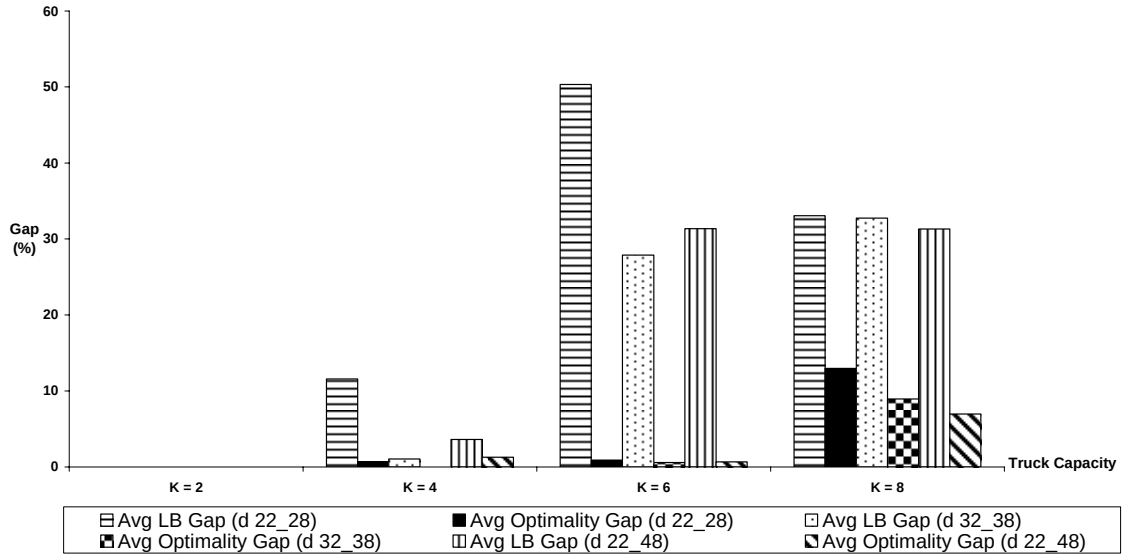

Figure 5.27: Numerical results for Heuristic 4 for $n = 10$ where $p_j \in [1,35]$

As observed in Figures 5.28 and 5.29, an increase in the number of jobs causes a decrease in *LB gaps* because the effect of the first trip decreases.

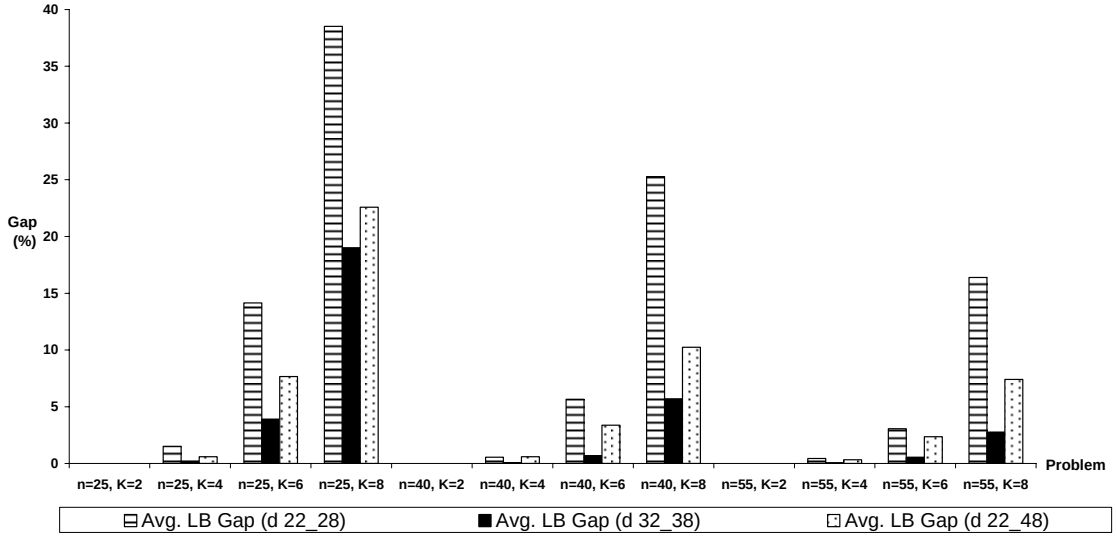

Figure 5.28: Numerical results for Heuristic 3 for $n = 25, 40, 55$ where $p_j \in [1,35]$

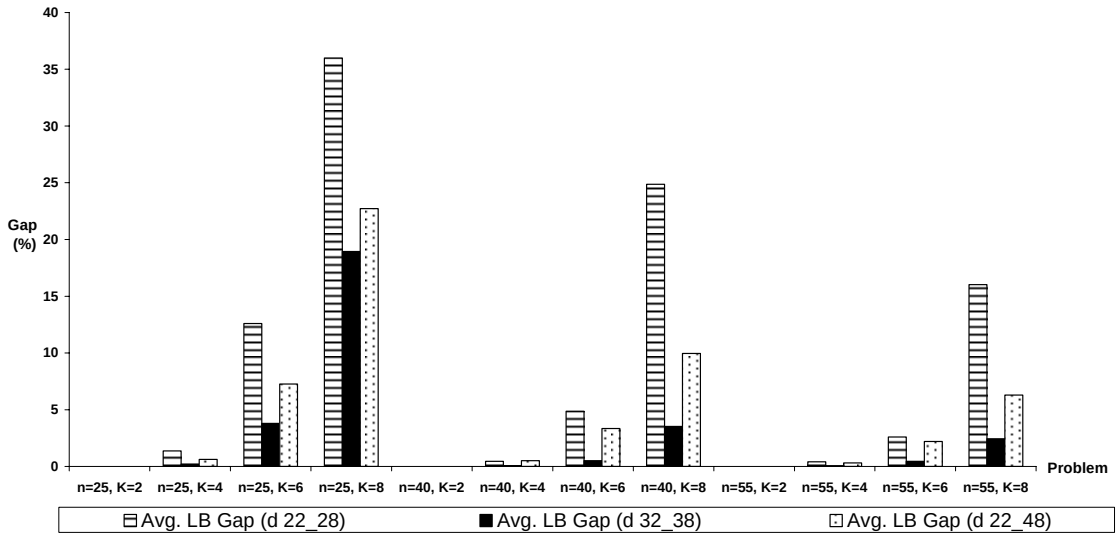

Figure 5.29: Numerical results for Heuristic 4 for $n = 25, 40, 55$ where $p_j \in [1, 35]$

5.2.3. Numerical Results and Discussions for the Problem with No Wait Constraint

As explained in detail in Chapter 4, in this problem, the truck does not wait for the completion time of a job at a plant. It leaves the plants it visits on any trip as soon as it arrives there. One important point to mention for this problem is that for all processing time ranges *LB gap* increases or remains the same as the capacity of the truck increases (this pattern can be seen in the result tables in Appendix E). This result follows from the fact that lower bound decreases with increasing capacity. As the capacity increases, the maximum number of jobs that can be transported on each trip increases and hence the number of jobs that can be transported in the trips more to the beginning of the schedule increases. This may cause a decrease in the number of required trips. Therefore, the value of the lower bound decreases.

Processing Time Range 1:

In this range, Heuristics 5 and 6 yield the same total completion times. For all plant distance ranges and (n, K) pairs *LB gap* values are very close to 0. The only exception is observed for distance range [22,48] and $(n = 10, K = 8)$. But as seen in Figure 5.30, proposed heuristics yield the optimal results even for this exception.

In Figure 5.31, it can be observed that average *LB gaps* are very small. In addition to that, an increase in the mean of the plant distance range causes a decrease in average *LB gaps*. Actually, in this problem for all processing time ranges, an increase in the mean of the distance range causes a decrease in the average *LB gap*. This result has two possible reasons. The first reason is that when the mean plant distance is large, for processing time ranges with small means, jobs are produced only by using the closer plants and the truck usually makes fully loaded trips in compliance with the schedule in the lower bound. The second reason is that for processing times with large means, since the heuristic method may require the usage of farther plants even for the first trip, the schedule generated by the heuristic and the lower bound will be similar and hence *LB gap* will decrease.

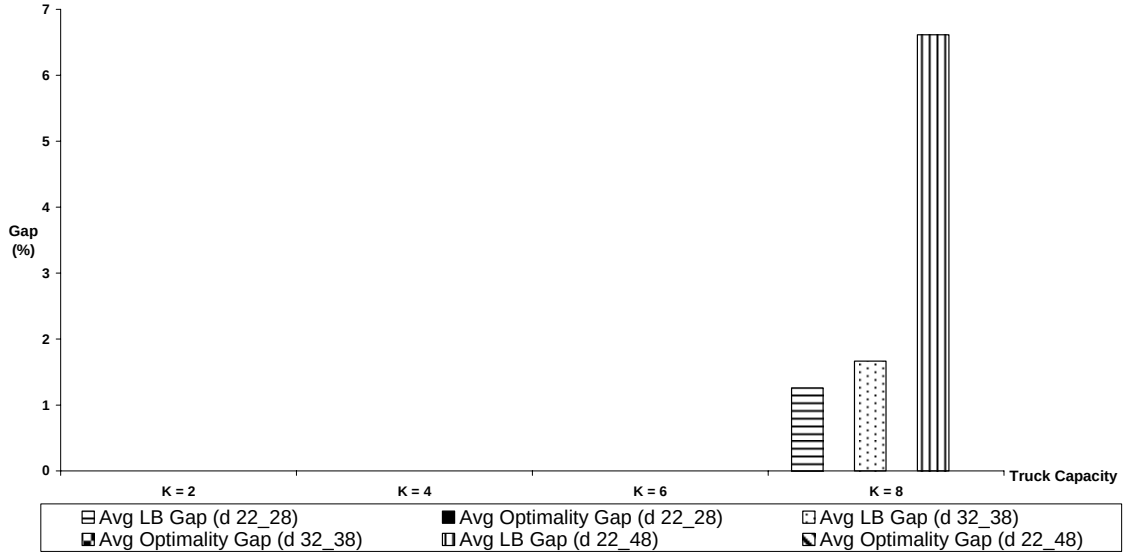

Figure 5.30: Numerical results for Heuristic 5 for $n = 10$ where $p_j \in [1,11]$

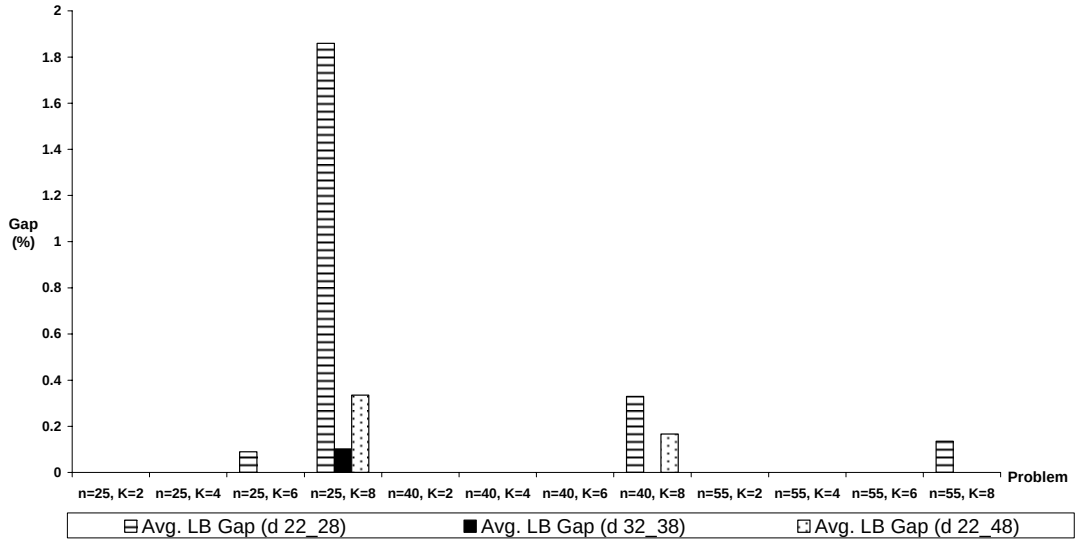

Figure 5.31: Numerical results for Heuristic 5 for $n = 25, 40, 55$ where $p_j \in [1, 11]$

Processing Time Range 2:

In this processing time range, for 10 jobs, it is seen in Figure 5.32 that there exists some instances with *LB gap* values larger than 10%. The results obtained by solving MIP-3 show that proposed heuristics generate optimal or near-optimal solutions for this problem. This implies the poor performance of Lower Bound 5. One possible reason for this performance of the lower bound is that trip completion times are calculated as if each trip is made to the closest plant. Despite this assumption, to be able to obtain a valid lower bound, the number of jobs on the first trip is calculated as if the first trip is made to the farthermost plant. In our heuristic methods, in the first trip, truck may choose to visit the farthermost plant and hence the number of jobs transported in the first trip would be the same in the heuristic method and the lower bound. However, in the heuristic methods, completion time of the first trip is calculated based on the distance of the farthermost plant instead of the closest plant. Hence, Lower Bound 5 yields a smaller total completion time than the heuristic methods.

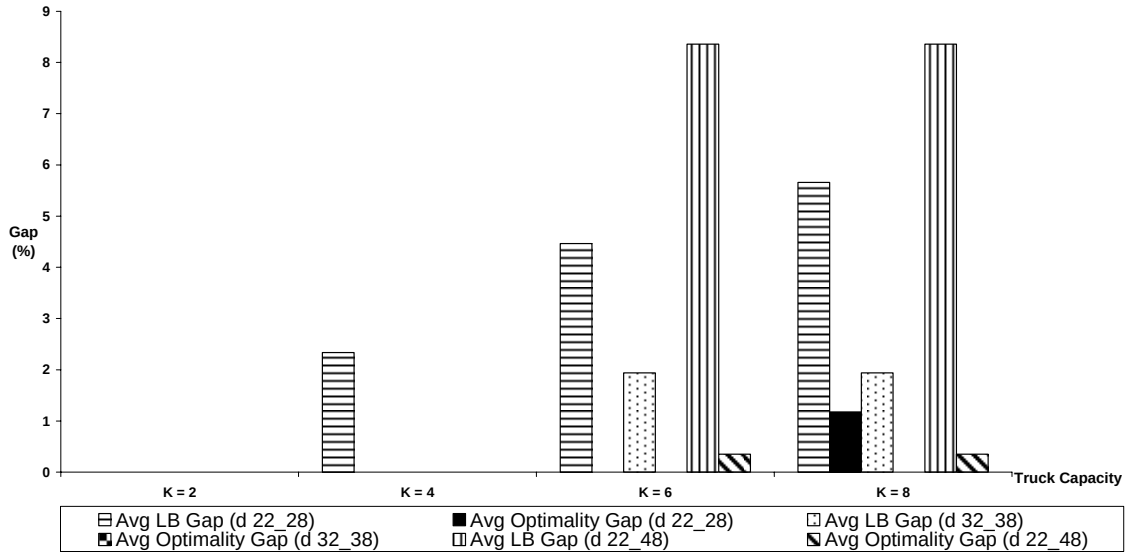

Figure 5.32: Numerical results for Heuristic 5 for $n = 10$ where $p_j \in [5,15]$

Figures 5.33 and 5.34 show the comparison of the average *LB gaps* for different plant distance ranges. As can be seen in these figures, an increase in the mean of the plant distances causes a decrease in average *LB gaps*. The reasons for this result were explained in Processing Time Range 1.

In the CPU times column of Table E.4 in Appendix E, it is possible to see some recurring values. The reason for this is as follows. The first instance of the problem with $(n = 10, K = 8)$ is solved optimally. In the optimal solution it was seen that in the first trip 4 jobs and in the second trip 6 jobs were carried. Although the truck capacity is 8, maximum 6 jobs were carried on each trip. Hence, the optimal solution for the first instance of the problem with $(n = 10, K = 6)$ is the same as the optimal solution of the problem with $(n = 10, K = 8)$ since all the parameters except the truck capacity are the same. Therefore, the CPU time obtained in solving the first instance of the problem with $(n = 10, K = 8)$ is applied

to the first instance of the problem with $(n=10, K=6)$. This procedure is used for all applicable instances in the problem with no wait constraint and in the general problem.

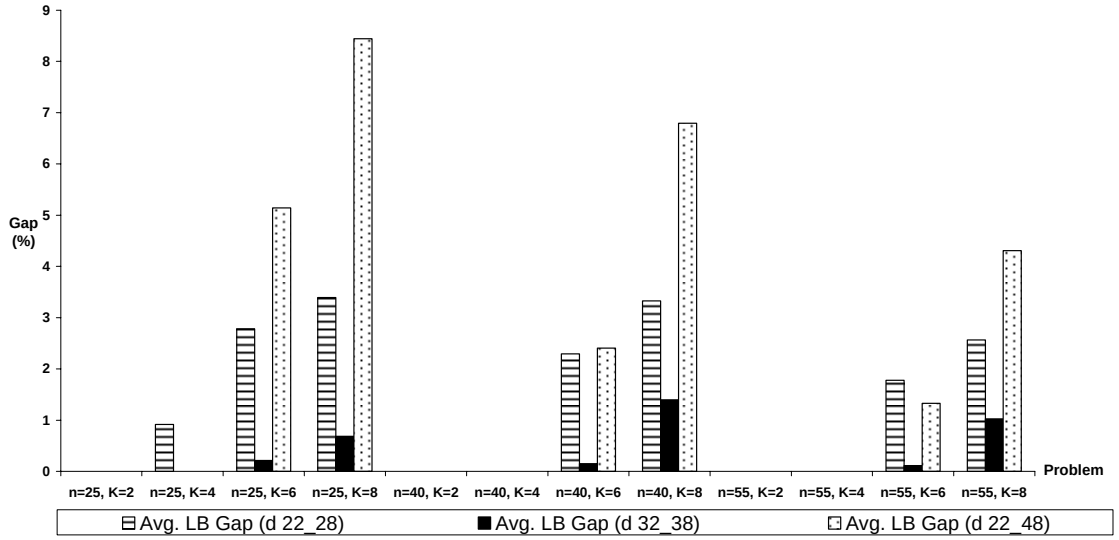

Figure 5.33: Numerical results for Heuristic 5 for $n=25,40,55$ where $p_j \in [5,15]$

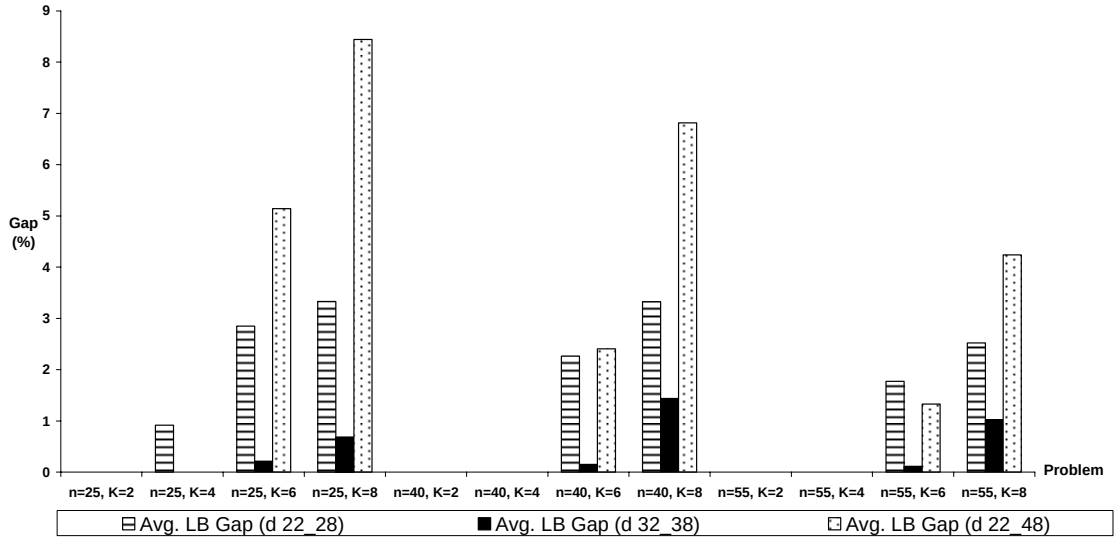

Figure 5.34: Numerical results for Heuristic 6 for $n=25,40,55$ where $p_j \in [5,15]$

As seen in Figures 5.33 and 5.34, an increase in the range of plant distances causes an increase in average *LB gaps*. This result is in compliance with our expectations because in Lower Bound 5, the number of jobs transported in each trip is calculated as if the truck is visiting farther plants but the completion times of the trips are calculated as if the truck is visiting the closest plant. As the difference between the closest and farthermost plant increases, lower bound becomes loose and hence *LB gap* increases. This result holds for each processing time range.

Processing Time Range 3:

In this range, we observe *LB gap* values larger than 10%. In order to determine the cause of these large gaps, we tried to obtain the optimal solution for the problems with $n = 10$. However, in 5 hours, we could not get the optimal results. Therefore, in order to be able to prove the performance of the proposed heuristic methods, we increased the resource limit to 41.67 hours (i.e. 150000 seconds). Even with this resource limit, we could not obtain the optimal solution for some instances. Therefore, instead of giving a bar chart showing the average gaps, we give the *LB* and *optimality gaps* for the instances for which the optimal solutions could be obtained in Tables 5.5, 5.6 and 5.7. In Tables E.7, E.8 and E.9 in Appendix E, the best integer solutions obtained in 150000 seconds are given with a (*) mark.

In Figures 5.35 and 5.36, comparisons of the *LB* gaps for different distance ranges are given for Heuristic 5 and Heuristic 6, respectively.

Table 5.5: *LB* and *optimality gaps* for Heuristic 5 and Heuristic 6 for $n = 10$ and $d_i \in [22, 28]$

		LB Gap (%)		Optimality Gap (%)	
n	K	**Heuristic 5**	**Heuristic 6**	**Heuristic 5**	**Heuristic 6**
10	**4**	8.40	2.14	6.14	0.00
		16.20	16.20	8.50	8.50
		9.02	9.02	0.00	0.00
		9.69	9.69	6.50	6.50
		8.40	8.40	---	---
10	**6**	12.57	6.07	6.14	0.00
		16.20	16.20	8.50	8.50
		9.02	9.02	0.00	0.00
		9.69	9.69	6.50	6.50
		8.40	8.40	---	---
10	**8**	12.57	6.07	6.14	0.00
		16.20	16.20	8.50	8.50
		9.02	9.02	0.00	0.00
		9.69	9.69	6.50	6.50
		8.40	8.40	---	---

Table 5.6: *LB* and *optimality gaps* for Heuristic 5 and Heuristic 6 for $n = 10$ and $d_i \in [32, 38]$

		LB Gap (%)		Optimality Gap (%)	
n	K	**Heuristic 5**	**Heuristic 6**	**Heuristic 5**	**Heuristic 6**
10	**6**	1.62	1.62	0.00	0.00
		12.38	8.21	6.50	2.56
		13.26	9.09	6.53	2.61
		9.03	1.62	7.29	0.00
		9.03	4.86	7.90	3.78
10	**8**	1.62	1.62	0.00	0.00
		12.38	8.21	6.50	2.56
		13.26	9.09	6.53	2.61
		9.03	1.62	7.29	0.00
		9.03	4.86	7.90	3.78

Table 5.7: *LB* and *optimality gaps* for Heuristic 5 and Heuristic 6 for $n = 10$ and $d_i \in [22, 48]$

		LB Gap (%)		Optimality Gap (%)	
n	K	**Heuristic 5**	**Heuristic 6**	**Heuristic 5**	**Heuristic 6**
10	**4**	1.80	1.80	---	---
		12.26	11.48	0.69	0.00
		14.29	14.29	0.00	0.00
		3.89	3.89	---	---
		0.00	0.00	0.00	0.00
10	**6**	1.88	1.88	0.00	0.00
		24.06	24.06	---	---
		19.05	19.05	0.00	0.00
		4.05	4.05	0.00	0.00
		0.00	0.00	0.00	0.00
10	**8**	1.88	1.88	0.00	0.00
		24.06	24.06	---	---
		19.05	19.05	0.00	0.00
		4.05	4.05	0.00	0.00
		0.00	0.00	0.00	0.00

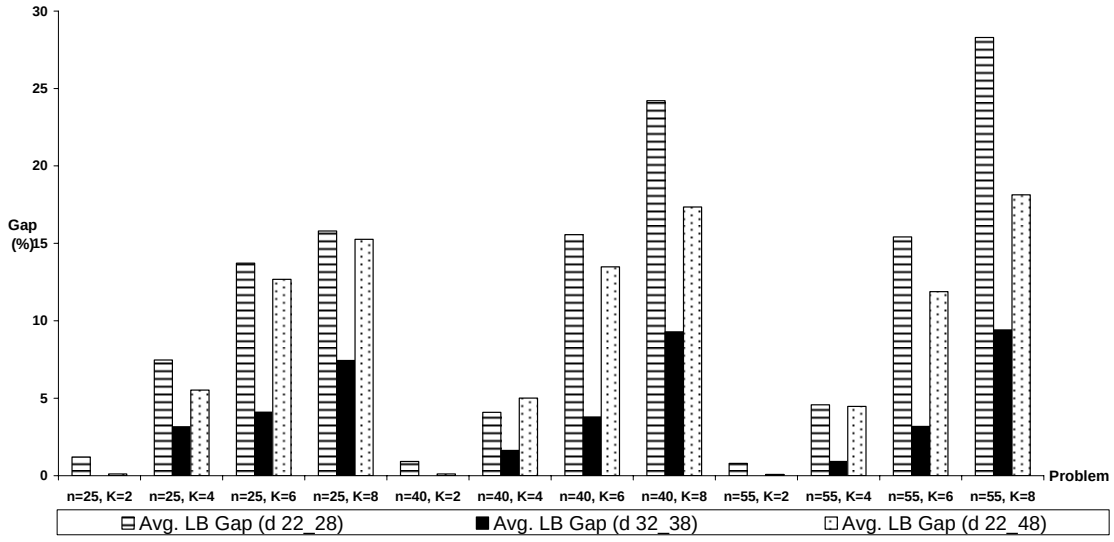

Figure 5.35: Numerical results for Heuristic 5 for $n = 25, 40, 55$ where $p_j \in [25, 35]$

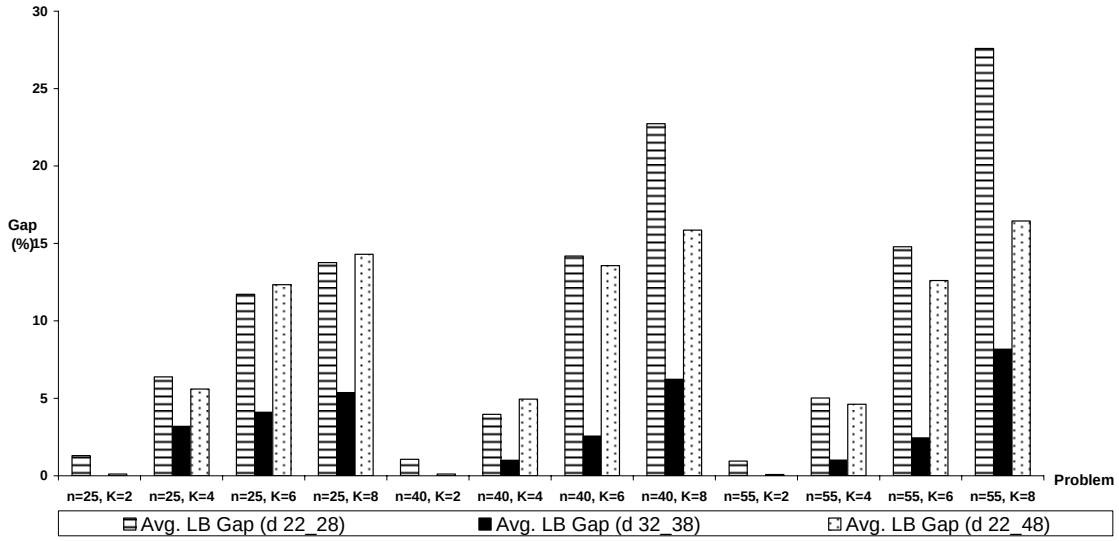

Figure 5.36: Numerical results for Heuristic 6 for $n = 25, 40, 55$ where $p_j \in [25, 35]$

Processing Time Range 4:

Since the range of the processing times is large, Lower Bound 5 may not yield tight lower bounds. That is because in the lower bound, while calculating the number of jobs to be carried on each trip, the processing times of the smallest jobs are used. However, in the proposed heuristics, as the jobs with smaller processing times are assigned to some plant, the jobs having larger processing times are left and hence the number of jobs to be transported on each trip decreases. This deviation from the lower bound causes *LB gaps* to be large. In Figures 5.37 and 5.38, it can be seen for 10 jobs that proposed heuristics yield either optimal or near-optimal results.

In Figures 5.39 and 5.40, it is observed that as the number of jobs to be transported increases, average *LB gap* decreases. This is related to the large range of processing times. As the number of jobs increases, the number of jobs with smaller processing times also increases. When there are more jobs with smaller processing times, more jobs complete

processing earlier and are transported in the beginning of the schedule. Therefore, the number of jobs transported in the first trips increases, decreasing the *LB gap*.

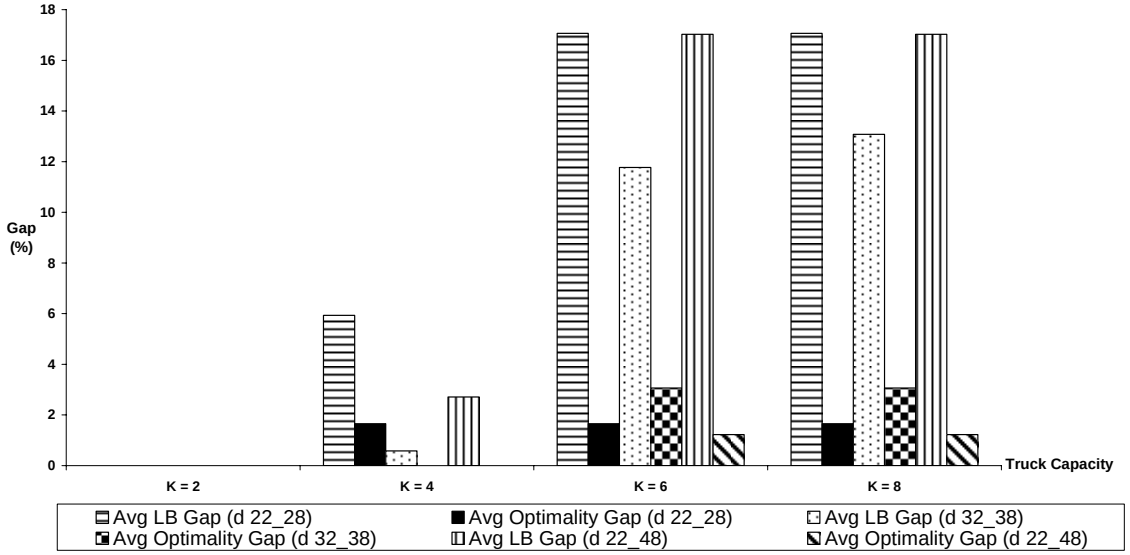

Figure 5.37: Numerical results for Heuristic 5 for $n = 10$ where $p_j \in [1,35]$

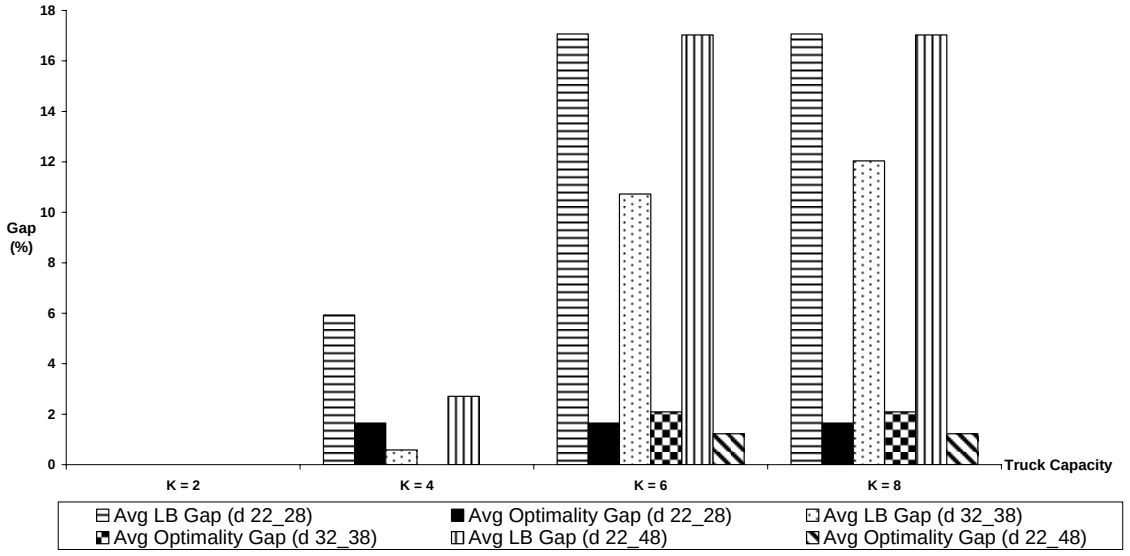

Figure 5.38: Numerical results for Heuristic 6 for $n = 10$ where $p_j \in [1,35]$

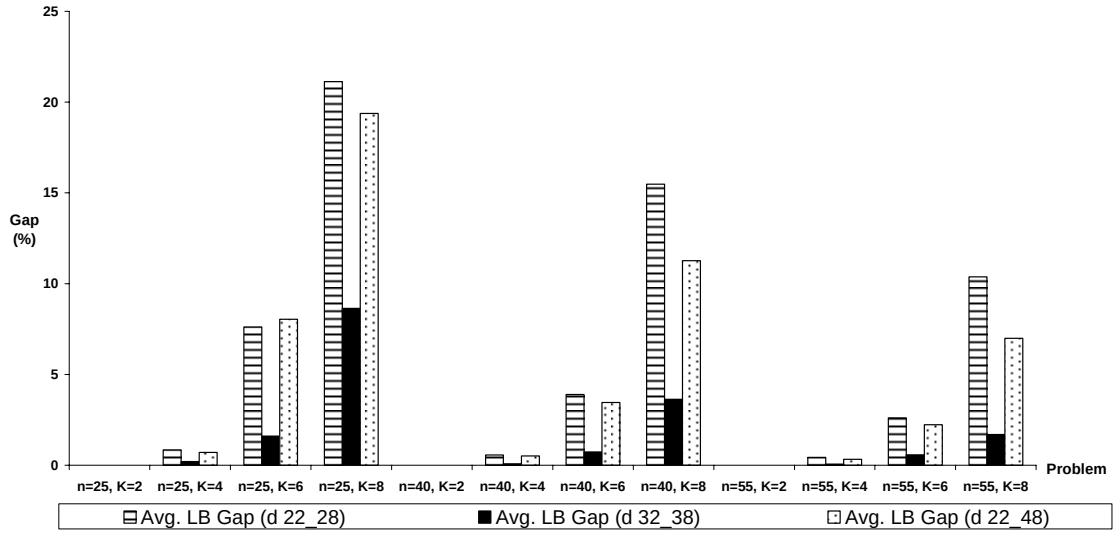

Figure 5.39: Numerical results for Heuristic 5 for $n = 25, 40, 55$ where $p_j \in [1, 35]$

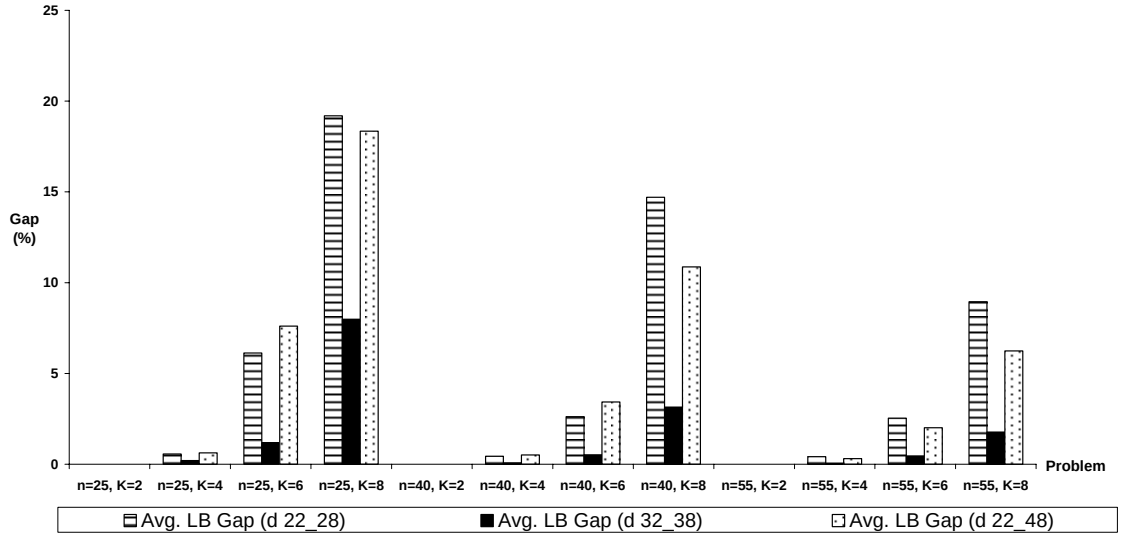

Figure 5.40: Numerical results for Heuristic 6 for $n = 25, 40, 55$ where $p_j \in [1, 35]$

5.2.4. Numerical Results and Discussions for the General Problem

As mentioned in Chapter 4, while generating the lower bound for this problem, it was supposed that the truck leaves the first visited plant either as soon as it arrives there or at the process completion time of the first job. For the rest of the trips, completion times were calculated as if the truck always visits the closest plant and leaves it immediately and fully loaded. As truck capacity increases, the number of jobs transported in each trip increases and hence more jobs complete their transportation in earlier trips. Also, the number of trips required to transport all jobs, i.e. $\left\lceil \frac{n}{K} \right\rceil$ may decrease. Therefore, as truck capacity increases, the lower bound for the total completion time decreases. This drawback of Lower Bound 6 causes the *LB gap* to increase as the capacity of the truck increases. Also, as the mean of the processing times increases, the possibility of producing K jobs until the arrival of the truck decreases. Hence, *LB gap* increases.

In this problem, as the mean of the plant distance range increases, *LB gap* decreases. As explained in Sections 5.2.1 and 5.2.2, as the mean distance increases, it becomes more probable that the truck leaves the plant either immediately or after waiting a small amount of time and fully loaded. Since this is the condition Lower Bound 6 is based on, *LB gap* becomes smaller. The effect of an increase in the range of plant distances is not as obvious as the effect of an increase in the mean. If the distance of the closest plant in range [22,48] is smaller than the one in range [32,38], obtained *LB gap* is larger. Conversely, if larger, obtained *LB gap* is smaller.

As mentioned in Chapter 4, since all the problems in the previous sections are special cases of this problem, all methods developed for them can be used to obtain a solution for the main problem. However, in this section, we only investigate the performance of Heuristic 7. The test results with Heuristic 7 can be seen in Appendix F.

Processing Time Range 1:

Since the processing times of the jobs are small when compared to the plant distances, in general, all jobs are produced in the closest plant and the truck finds K jobs already produced when it arrives at the plant. However, when the capacity of the truck increases, it may not be able to transport K jobs on the first trip by leaving the plant immediately. If the truck waits at the plant for the completion of a job in order to transport more jobs, the completion times of subsequent trips may be delayed. If it leaves the plant immediately by transporting fewer jobs, the number of jobs transported on the later trips increases. Any of these cases will result in *LB gaps* greater than 0. In Figure 5.41, average *LB gap* larger than 10% is observed for $n = 10$ and $K = 8$. As seen in this figure, Heuristic 7 generates the optimal solution.

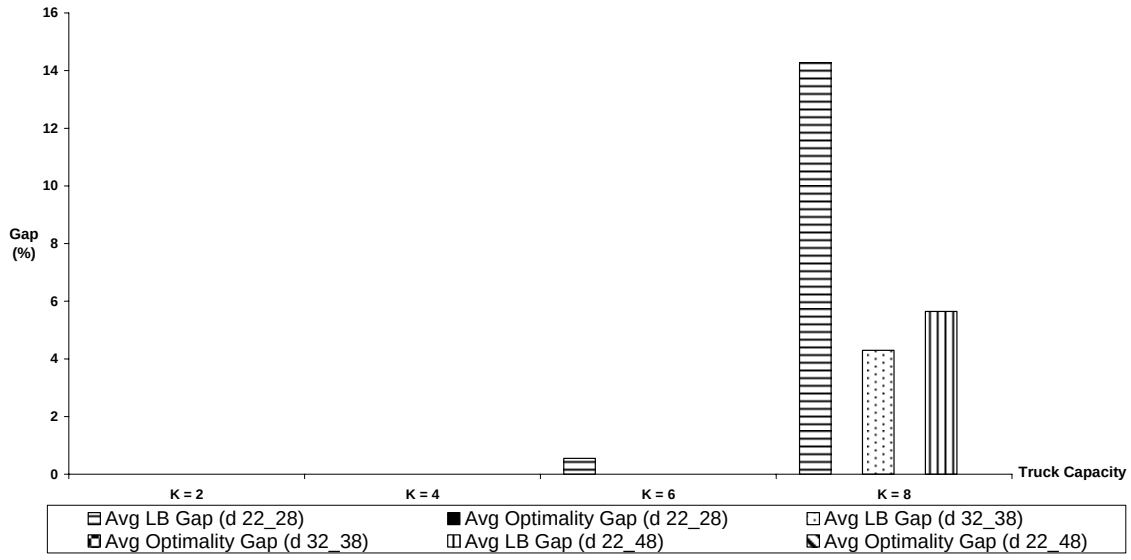

Figure 5.41: Numerical results for Heuristic 7 for $n = 10$ where $p_j \in [1,11]$

As explained in the previous paragraph, the main reason for large *LB gaps* is not being able to transport K jobs on the first trip from the closest plant. However, as n increases, the number of trips required to transport all jobs increases and hence the effect of

the first trip decreases. Therefore, for the same K value, as n increases, *LB gap* decreases. This effect of the increase in the number of jobs can be observed in Figure 5.42.

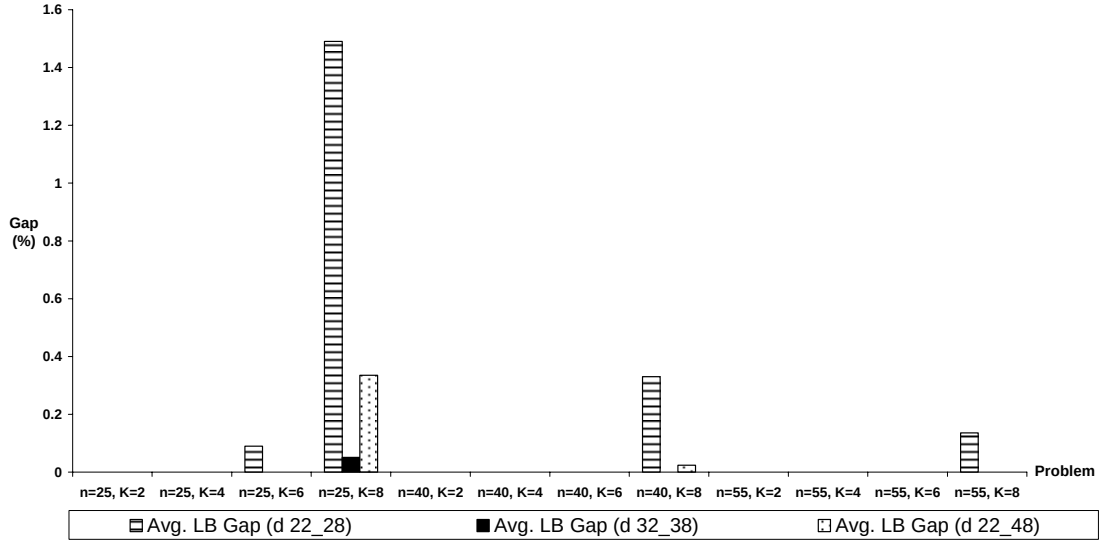

Figure 5.42: Numerical results for Heuristic 7 for $n = 25, 40, 55$ where $p_j \in [1, 11]$

Also, it should be noted that for this processing time range, Heuristic 7 performs better than the heuristics proposed for the previous problems.

Processing Time Range 2:

In this range, since $p_{\max} < d_{\min}$, in the calculation of the lower bound, the truck is supposed to leave the first plant as soon as it arrives there. Hence, the lower bound values found for this range are the same with the ones found for processing time range 1. However, since the processing times for range 2 are larger, the probability of transporting K jobs on the first trip is smaller. Therefore, observed *LB gaps* are larger when compared to the previous range (at most 43.40%). In Figure 5.43, it is shown for 10 jobs that Heuristic 7 generates optimal or near optimal results with at most 0.89% *optimality gap*.

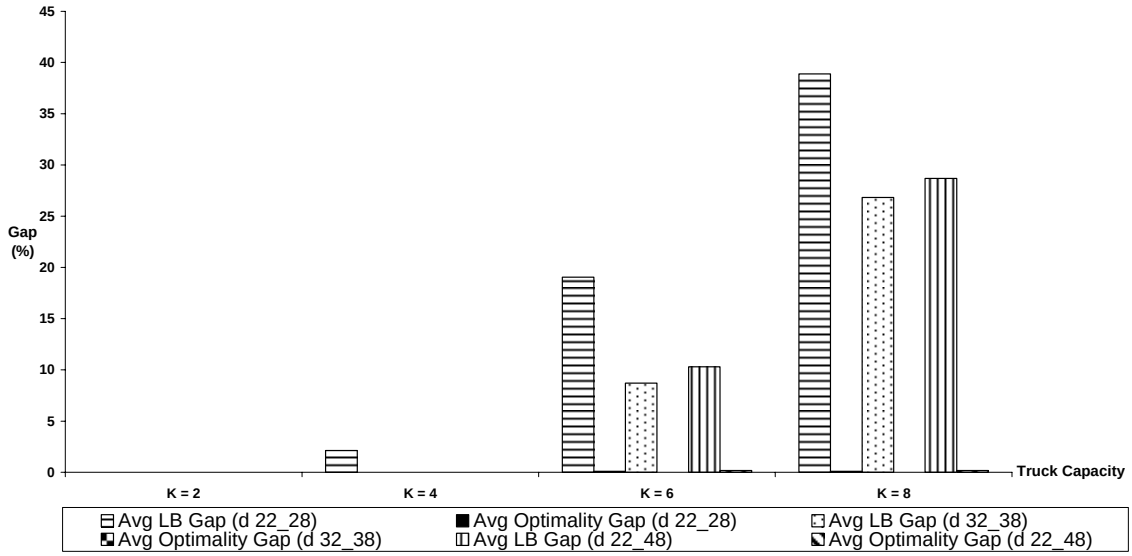

Figure 5.43: Numerical results for Heuristic 7 for $n = 10$ where $p_j \in [5,15]$

Not being able to transport K jobs on the first trip without waiting for the completion of a job at the plant is the cause for the deviation from the lower bound. Hence, as n increases, for the following trips, truck may be able to transport K jobs without waiting at the plant and deviation from the lower bound will decrease. This can be observed in Figure 5.44.

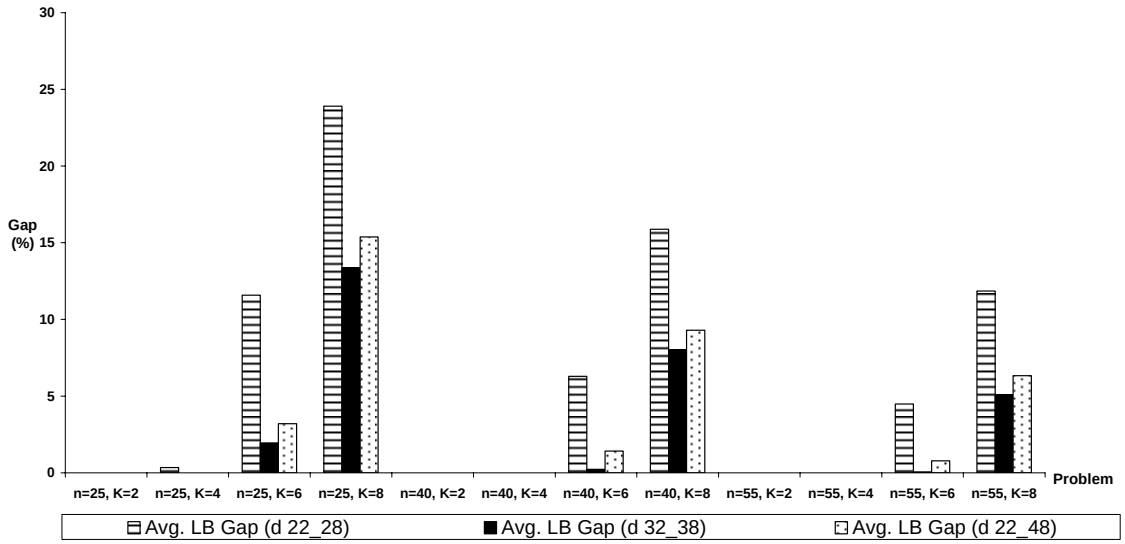

Figure 5.44: Numerical results for Heuristic 7 for $n = 25, 40, 55$ where $p_j \in [5,15]$

Processing Time Range 3:

The largest processing time values are observed in this range and hence, the probability of sending the truck fully loaded without waiting at a plant is smallest. Therefore, *LB gaps* observed in this range are largest (as high as 146.42%). As explained for the previous ranges, an increase in n decreases *LB gap* and hence largest gaps occur for 10 jobs. To show that our heuristic methods yield optimal or near optimal solutions for these largest gap instances, we solve MIP-4 to obtain optimal solutions. However, even in 55.55 hours (200000 sec), we could not obtain the optimal solution for some instances (the best integer values obtained in 200000 seconds are given with a (*) mark in Table F.7). Hence, instead of bar charts showing average *LB* and *optimality gaps*, the percentage gaps are given in Tables 5.8, 5.9 and 5.10 for 10 jobs. In these tables, it can be seen that proposed heuristic yields optimal or near-optimal results.

Table 5.8: *LB* and *optimality gaps* for Heuristic 7 for $n = 10$ and $d_i \in [22, 28]$

		LB Gap (%)	**Optimality Gap**
n	K	**Heuristic 7**	**Heuristic 7**
10	**2**	17.56	0.77
		17.79	---
		17.71	---
		17.82	0.00
		16.67	---
10	**4**	53.21	0.00
		65.22	4.03
		59.67	0.00
		56.09	1.11
		56.09	1.18
10	**6**	96.98	0.00
		111.73	4.03
		103.92	0.00
		100.69	1.11
		100.69	1.18
10	**8**	129.81	0.00
		146.42	4.03
		136.71	0.00
		134.13	1.11
		134.13	1.18

Table 5.9: *LB* and *optimality gaps* for Heuristic 7 for $n = 10$ and $d_i \in [32, 38]$

		LB Gap (%)	Optimality Gap (%)
n	K	**Heuristic 7**	**Heuristic 7**
10	**2**	6.94	0.00
		8.33	0.00
		9.09	0.00
		7.87	0.00
		6.94	0.00
10	**4**	35.03	0.00
		37.42	0.00
		38.72	0.00
		36.57	0.00
		33.80	0.00
10	**6**	73.61	0.00
		76.68	0.00
		82.68	2.43
		74.21	0.00
		72.02	0.00
10	**8**	102.55	0.00
		106.13	0.00
		113.13	2.43
		103.24	0.00
		100.69	0.00

Table 5.10: *LB* and *optimality gaps* for Heuristic 7 for $n = 10$ and $d_i \in [22, 48]$

		LB Gap (%)	Optimality Gap
n	K	**Heuristic 7**	**Heuristic 7**
10	**2**	4.58	0.00
		11.51	0.00
		16.07	0.00
		7.21	0.00
		5.13	0.00
10	**4**	32.08	0.00
		47.67	0.00
		58.33	0.88
		39.04	0.00
		30.77	0.00
10	**6**	70.54	1.49
		89.86	0.00
		103.57	0.88
		78.38	0.00
		68.13	0.00
10	**8**	98.96	1.49
		121.51	0.00
		137.50	0.88
		108.11	0.00
		96.15	0.00

In Figure 5.45, the *LB gaps* for different plant distance ranges are given. Since the processing times are large, lower bound is very loose as it supposes the truck makes all trips to the closest plant and leaves the plant fully loaded.

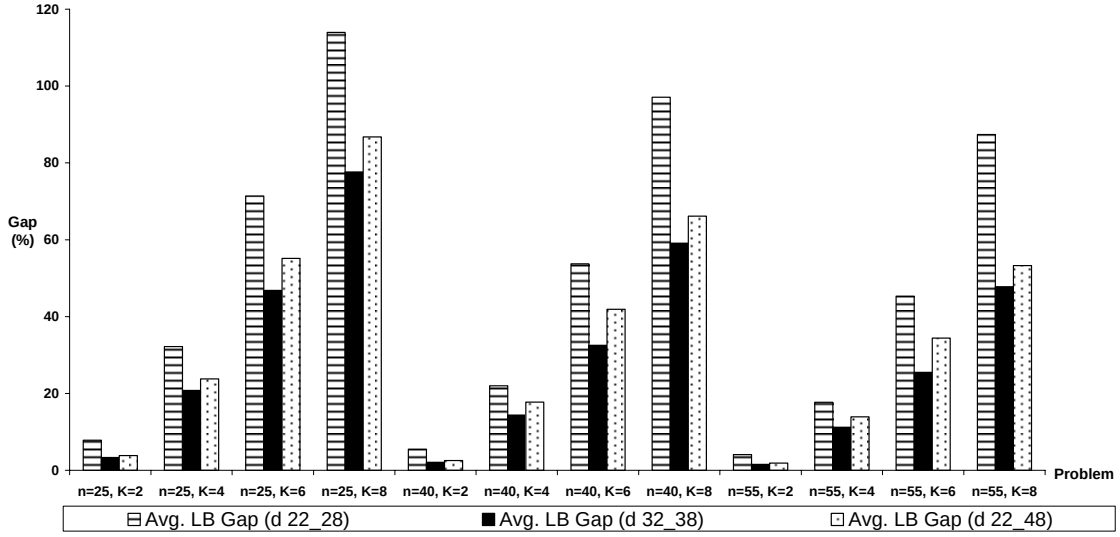

Figure 5.45: Numerical results for Heuristic 7 for $n = 25, 40, 55$ where $p_j \in [25, 35]$

Processing Time Range 4:

Since the jobs in the beginning of the job sequence have small processing times, the number of jobs transported on the first trips is greater and hence *LB gaps* are smaller when compared to processing time range 3. Also, since the jobs at the end of the job sequence have larger processing times, the truck may have to visit a farther plant and hence *LB gaps* are larger when compared to the first two ranges.

As the number of jobs increases, *LB gap* decreases. Hence, the largest gaps are observed for 10 jobs. The maximum *LB gap* obtained is 74.52 % and the maximum *optimality gap* obtained is 2.53%. The percentage gaps for 10 jobs are shown in Figure 5.46 for different plant distance ranges.

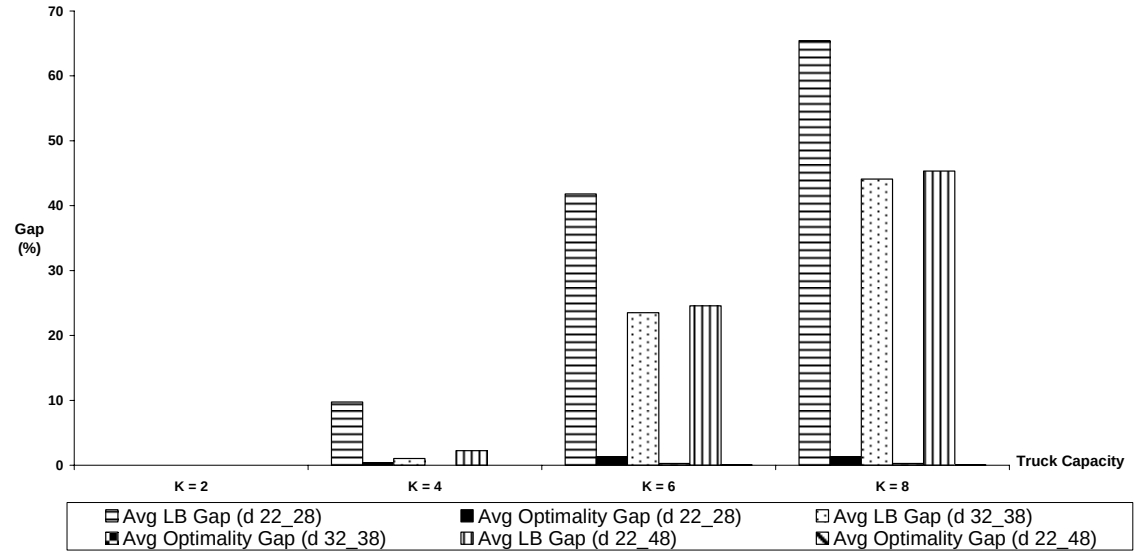

Figure 5.46: Numerical results for Heuristic 7 for $n = 10$ where $p_j \in [1,35]$

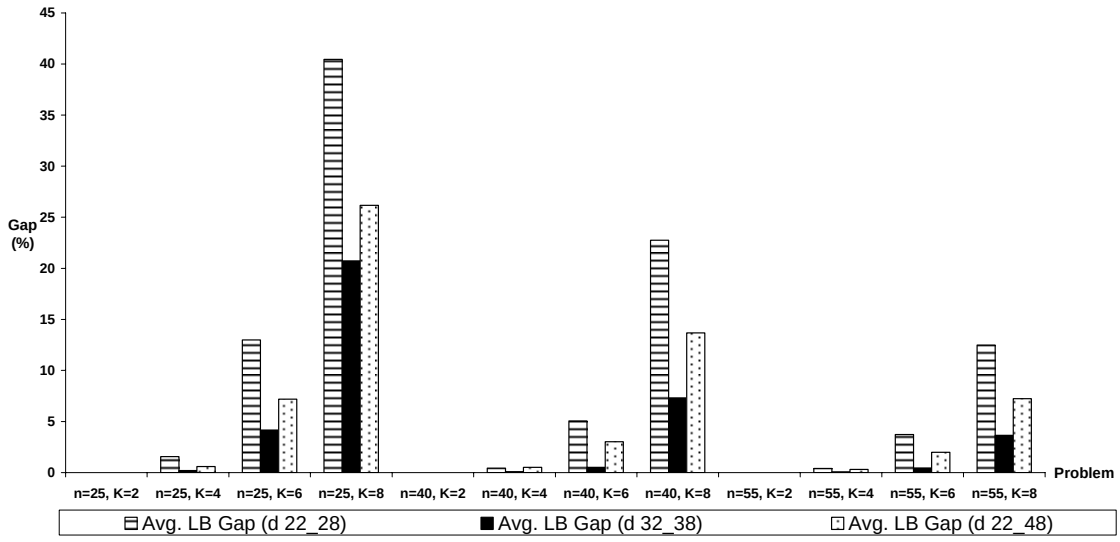

Figure 5.47: Numerical results for Heuristic 7 for $n = 25, 40, 55$ where $p_j \in [1,35]$

In Figure 5.47, average *LB gaps* of different plant distance ranges are given. The effects of increases in the mean and range of plant distances are explained in the beginning of this section.

5.2.5. Comparison of Proposed Methods

The special cases of the general problem are selected from common practical applications. There are two main strategies used in these applications:

1. Send the trucks fully loaded (one partial trip may occur as explained before)
2. The truck leaves the plant as soon as it arrives there

The first strategy includes Last Trip Partial Problem and Fully Loaded Trips Problem. The second strategy is addressed by No Wait Problem. Especially in the example of the leading soft drink manufacturer these two strategies have been applied for a long time. Any change in these strategies may complicate the operations and also requires education of the personnel. Hence, most of the companies do not want to replace these existing strategies with new ones. Therefore, in order to be able to suggest one of these simple strategies for the general problem, we assess the performance of the proposed heuristics on the general problem.

To determine the quality of proposed methods on the general problem, we compare the results of the proposed methods for special cases and the results for the general problem with respect to two aspects:

1. The optimal results
2. The heuristic results

Comparisons of the optimal results are done only for 10 jobs since it was not possible to obtain optimal solutions for larger problems even in 50000 seconds. The percentage gap between the optimal result of the special case and optimal result of the general problem is calculated as follows:

$$Opt\ Gap = \frac{Optimal\ Solution\ of\ Special\ Case - Optimal\ Solution\ of\ General\ Problem}{Optimal\ Solution\ of\ General\ Problem} \cdot 100$$

For each special case, the average *opt gaps* are calculated and the one yielding the smallest average is chosen.

To compare the heuristic results, for each special case, the best (i.e. the smallest) heuristic value is chosen. These chosen values are then compared to the heuristic results obtained by Heuristic 7. The percentage gap is calculated as follows:

$$HR\ Gap = \frac{Best\ Heuristic\ Value\ of\ Special\ Case - Heuristic\ Value\ of\ General\ Problem}{Heuristic\ Value\ of\ General\ Problem} \cdot 100$$

HR gaps are determined for $n = 10, 25, 40, 55$ and average *HR gap* is calculated for each special case. The problem yielding the smallest *HR gap* is chosen. In compliance with our expectations, the problem chosen as a result of optimal solution comparison is the same as the problem chosen as a result of heuristic solution comparison. The problems yielding the best average gaps are listed in Table 5.11 for each processing time and plant distance range.

It is obvious that the total completion times obtained by the heuristics proposed for fully loaded trips problem will always be better than or the same as the ones obtained by the heuristics for last trip partial problem. However, since the policy of sending the last trip partial is easier, we choose that policy when its results are the same with fully loaded trips problem.

For processing time range $[25, 35]$, as explained in Section 5.2.3 and 5.2.4, we could not obtain the optimal results for some of the instances even with 10 jobs. Hence an average *opt gap* is not given for this problem.

Table 5.11: Performance assessment of the methods of special cases for the general problem

	Chosen Problem	**Avg. Opt. Gap (%)**	**Avg. HR Gap (%)**
$p_j \in [1,11],\ d_i \in [22,28]$	No Wait	0.67	0.19
$p_j \in [1,11],\ d_i \in [32,38]$	Last Trip Partial	0.28	0.07
$p_j \in [1,11],\ d_i \in [22,48]$	Last Trip Partial	0.32	0.08
$p_j \in [5,15],\ d_i \in [22,28]$	Full Loaded Trips	2.86	1.13
$p_j \in [5,15],\ d_i \in [32,38]$	No Wait	0.88	0.71
$p_j \in [5,15],\ d_i \in [22,48]$	No Wait	1.12	0.75
$p_j \in [25,35],\ d_i \in [22,28]$	No Wait	---	1.17
$p_j \in [25,35],\ d_i \in [32,38]$	No Wait	---	2.21
$p_j \in [25,35],\ d_i \in [22,48]$	No Wait	---	2.55
$p_j \in [1,35],\ d_i \in [22,28]$	No Wait	1.15	0.96
$p_j \in [1,35],\ d_i \in [32,38]$	No Wait	1.79	0.89
$p_j \in [1,35],\ d_i \in [22,48]$	No Wait	1.20	0.70

In Table 5.11 it can be seen that when the processing times are small compared to the plant distances, sending the truck fully loaded yields smallest gaps. However, as the processing times increase, waiting for the truck to be fully loaded becomes disadvantageous. Hence, leaving the plant immediately yields smallest gaps.

Chapter 6

CONCLUSION

In this thesis, we studied the problem of integrated scheduling of production and distribution operations of a manufacturer serving a single customer area from multiple identical production plants at different geographical points. The products are transported from the plants to the customer area by a single capacitated truck. The objective is to minimize the total completion time of the jobs. The completion time of a job is defined as the time it reaches at the customer area.

We considered both this general problem and four special cases motivated by common practical applications. The first special case considers a setting where the assignment of the jobs to the plants is given. A pseudo-polynomial dynamic programming is proposed for that problem.

The soft drink manufacturer that motivates our problem and many other companies tend to be under the impression that they utilize truck capacity better when they send the trucks fully loaded. In addition to that, fully loaded trucks are preferred in some applications to prevent the damage to the products that may be caused when they are shaken or scattered around in the truck. To address this preference, the second special case considers the problem where the truck is fully loaded in each trip except the last one. The last trip may be partially loaded if the number of jobs demanded by the customer is not divisible by the truck capacity. The third special case again has the constraint of trips' being fully loaded; however, it is not

known in this problem which trip would be partially loaded. In the fourth special case, we suppose that the truck does not wait at the plant for the completion of a job. Instead, it takes the jobs that are already produced and leaves the plant immediately.

We proved that both the main problem and the last three special cases are NP-hard at least in the ordinary sense. We developed mixed integer programming (MIP) models for all these problems. Since the MIP models are able to provide optimal solutions only for small instances in a reasonable amount of time, we propose constructive heuristics to solve larger instances. For small instances, the optimal solutions obtained by these MIP models can be used to measure the quality of the proposed heuristics. However, since it is not possible to get optimal solutions for medium and large instances in reasonable CPU times, fast lower bounds are developed to facilitate the performance assessment of the heuristics. Based on extensive computational experimentation with the test problems generated, we observed that proposed heuristics provide optimal or near-optimal results in less than a second.

As mentioned before, three of the special cases consider two different strategies: sending full trucks and not waiting at the plant. Although these strategies are commonly applied, they may not yield the best results in terms of reduced total completion times. However, as observed in the soft drink company, companies are not willing to change these existing strategies. Thus, for different problems we determined which strategy would yield a smaller total completions time. It is shown that when the processing times of the jobs are small with respect to the plant distances, sending the last trip partial and the rest of the trips fully loaded yields the best results on the average. As the ratio of the mean of processing times to the mean of plant distances increases, waiting at the plant for the truck to be fully loaded becomes disadvantageous. Hence, for large processing times, it is a better strategy if the truck leaves the plant as soon as it arrives at there.

Finding stronger lower bounds for the problems studied in this thesis deserves attention as an item for future research. Another item relates to the shipment strategy. In light of the motivating practical application, we considered only direct shipments between the

plants and the customer area. As future research it would be interesting to consider milkruns among the plants. Also the problem of a multi-plant manufacturer serving multiple customer areas would be interesting to study. The problem with multiple capacitated trucks may also be studied.

BIBLIOGRAPHY

Chandra, P. and M.L. Fisher, "Coordination of Production and Distribution Planning", *European Journal of Operational Research*, 72 (1994), 503–517.

Chang, Y.-C. and C.-Y. Lee, "Machine Scheduling with Job Delivery Coordination", *European Journal of Operational Research*, 158 (2004), 470-487.

Chen, Z.-L. and G.L. Vairaktarakis, "Integrated Scheduling of Production and Distribution Operations", *Management Science*, 51 (2005), 614-628.

Chen, Z.-L. and G. Pundoor, "Order Assignment and Scheduling in a Supply Chain", *Operations Research*, 54 (2006), 555-572.

Chen, Z.-L., "Integrated Production and Outbound Distribution Scheduling: Review and Extensions", To appear in *Operations Research*, last review in 2008.

Cheng, T.C.E. and V.S. Gordon, "On Batch Delivery Scheduling on a Single Machine", *Journal of the Operational Research Society*, 45 (1994), 1211-1215.

Cheng, T.C.E., V.S. Gordon, and M.Y. Kovalyov, "Single Machine Scheduling with Batch Deliveries", *European Journal of Operational Research*, 94 (1996), 277-283.

Cheng, T.C.E., M.Y. Kovalyov, and B.M.-T. Lin, "Single Machine Scheduling to Minimize Batch Delivery and Job Earliness Penalties", *SIAM Journal on Optimization*, 7 (1997), 547-559.

Ertogral, K., S. D. Wu and L.I. Burke, "Coordination Production and Transportation Scheduling in the Supply Chain", *Technical Report#98T-010, Department of Industrial and Manufacturing Systems Engineering, Lehigh University*, (1998).

Fumero, F., C. Vercellis, "Synchronized Development of Production, Inventory, and Distribution Schedules", *Transportation Science*, 33 (1999), 330-340.

Graham, R.L., E.L. Lawler, J.K. Lenstra and A.H.G. Rinnooy Kan, "Optimization and Approximation in Deterministic Sequencing and Scheduling: A survey", *Annals of Discrete Mathematics*, 5 (1979), 287-326.

Hall, L. and D. Shmoys, "Approximation Schemes for Constrained Scheduling Problems" In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, 1989, 134-140.

Hall, L.A. and D.B. Shmoys, "Jackson's Rule for Single Machine Scheduling: Making a Good Heuristic Better", *Mathematics of Operations Research*, 17 (1992), 22-35.

Hall, N.G. and C.N. Potts, "Supply Chain Scheduling: Batching and Delivery", *Operations Research*, 51 (2003), 566-584.

Hall, N.G. and C.N. Potts, "The Coordination of Scheduling and Batch Deliveries", *Annals of Operations Research*, 135 (2005), 41-64.

Lee, C.-Y. and Z.-L. Chen, "Machine Scheduling with Transportation Considerations", *Journal of Scheduling*, 4 (2001), 3-24.

Lenstra, J.K., A.H.G. Rinnooy Kan and P. Brucker, "Complexity of Machine Scheduling Problems", *Annals of Discrete Mathematics*, 1 (1977), 343-362.

Li, C.-L., G. Vairaktarakis, and C.-Y. Lee, "Machine Scheduling with Deliveries to Multiple Customer Locations", *European Journal of Operational Research*, 164 (2005), 39-51.

Li, C.-L. and G. Vairaktarakis, "Coordinating Production and Distribution of Jobs with Bundling Operations", *IIE Transactions*, 39 (2007), 203-215.

Li, C.-L. and J. Ou, "Machine Scheduling with Pickup and Delivery", *Naval Research Logistics*, 52 (2005), 617-630.

Li, C.-L. and J. Ou, "Coordinated Scheduling of Customer Orders with Decentralized Machine Locations", *IIE Transactions*, 39 (2007), 899-909.

Potts, C.N., "Analysis of a Heuristic for One Machine Sequencing with Release Dates and Delivery Times", *Operations Research*, 28 (1980) 1436-1441.

Pundoor, G. and Z.-L. Chen, "Scheduling a Production-Distribution System to Optimize the Tradeoff between Delivery Tardiness and Distribution Cost", *Naval Research Logistics*, 52 (2005), 571-579.

Qi, X., "A Logistics Scheduling Model: Scheduling and Transshipment for Two Processing Centers", *IIE Transactions*, 38 (2006), 609-618.

Qi, X., "Coordinated Logistics Scheduling for In-House Production and Outsourcing", *IEEE Transactions on Automation Science and Engineering*, 5 (2008), 188-192.

Wang, G. and T.C.E. Cheng, "Parallel Machine Scheduling with Batch Delivery Costs", *International Journal of Production Economics*, 68 (2000), 177-183.

Woeginger, G.J. "Heuristics for Parallel Machine Scheduling with Delivery Times", *Acta Informatica*, 31 (1994) 503-512.

Woeginger, G.J. “A Polynomial-Time Approximation Scheme for Single-Machine Sequencing with Delivery Times and Sequence-Independent Batch Set-Up Times”, *Journal of Scheduling*, 1 (1998), 79 – 87.

Zdrzalka, S. “Approximation Algorithms for Single-Machine Sequencing with Delivery Times and Unit Batch Set-Up Times”, *European Journal of Operational Research*, 51 (1991), 199–209.

Zdrzalka, S. “Analysis of Approximation Algorithms for Single-Machine Scheduling with Delivery Times and Sequence Independent Setup Times”, *European Journal of Operational Research*, 80 (1995), 371 - 380.

Appendix A

Pseudocodes of the Lower Bound Generation Methods

A.1 Lower Bound 1

Algorithm of Lower Bound 1

Sort plants in nondecreasing distances and reindex
Sort jobs in nondecreasing processing times and reindex
Initialize *travelTime* and *arriveTime* arrays
Set $LB := 0$, $jobSent := 0$
Set d to closest plant distance
Set $partial := n - qK$
Set $reqTrip := \left\lceil \frac{n}{K} \right\rceil$
Set $minTrip := \left\lfloor \frac{n}{K} \right\rfloor$
Set *sumProc* to sum of processing times for the first K jobs
//Calculate *travelTime* for each trip
for each plant p_i
  for j from 0 to ($reqTrip - i$) (increase by m)
    Set $travelTime[i + j] := (2i\text{-}1)\ d$
  end for
end for
Set $leaveTime := \max(sumProc,\ d)$
for each trip t_k
  if (k-1) is divisible by m **and** ((k-1) / m) > 0 // if trip is the first trip of a round
    Increase *jobSent* by $m \cdot K$
    Set $numJobs := (k/m) \cdot K + \min(K, n\text{-}jobSent)$
    Set *sumProc1* to sum of processing times for the first *numJobs* jobs

Set *leaveTime* := max(*leaveTime* + 2*md* , *sumProc1*)
end if
Set *arriveTime*[*k*] := *leaveTime* + *travelTime*[*k*]
end for
for each trip t_k such that $k \neq reqTrip$
Increase *LB* by ($K \cdot$ *arriveTime*[*k*])
end for
if *n* is divisible by *K*
Increase *LB* by ($K \cdot$ *arriveTime*[*reqTrip*])
end if
Increase *LB* by (*partial* $\cdot$ *arriveTime*[*reqTrip*])
Return *LB*

A.2 Lower Bound 3

Algorithm of Lower Bound 3

Sort jobs and plants and reindex
Initialize *travelTime* and *arriveTime* arrays
Set $LB := 0$, $jobSent := 0$
Set *d* to closest plant distance
Set $partial := n - qK$
Set $reqTrip := \left\lceil \frac{n}{K} \right\rceil$
Set $minTrip := \left\lfloor \frac{n}{K} \right\rfloor$
//calculate *travelTime* for each trip
for each plant Pl_i
for *j* from 0 to (*reqTrip* – 1) (increase by *m*)
Set *travelTime*[*i* + *j*] := (2 *i* -1) *d*
end for
end for
if *partial* > 0
Set *sumProc* to sum of processing times for the first *partial* jobs
Set *leaveTime* := max (*sumProc*, *d*)

Set *minTrip1* $\coloneqq$ min ($2m$, *reqTrip*)
for each trip t_k such that $k \le$ *minTrip1*
if (k -1) is divisible by m **and** ((k -1) / m) > 0
Increase *jobSent* by ($(m-1)\cdot K$ + *partial*)
Set *numJobs* $\coloneqq$ *partial* + min(K , n -*jobSent*)
Set *sumProc1* to sum of processing times for the first *numJobs* jobs
Set *leaveTime* $\coloneqq$ max(*leaveTime* + $2md$, *sumProc1*)
end if
Set *arriveTime*[k] $\coloneqq$ *leaveTime* + *travelTime*[k]
end for
if *reqTrip* > $2m$
for each trip t_k such that $2m+1 \le k \le$ *reqTrip*
Calculate *arriveTime*[k] by *Algorithm Arrival Time Calculation*
end for
end if
else
Set *sumProc* to sum of processing times for the first K jobs
Set *leaveTime* $\coloneqq$ max (*sumProc*, d)
for each trip t_k
Calculate *arriveTime*[k] by *Algorithm Arrival Time Calculation*
end for
end if
if *partial* > 0
for each trip t_k such that $k \ne$ *reqTrip*
Increase *LB* by ($K \cdot$ *arriveTime*[k])
end for
Increase *LB* by (*partial* $\cdot$ *arriveTime*[*reqTrip*])
else
for each trip t_k
Increase *LB* by ($K \cdot$ *arriveTime*[k])
end for
end if

Algorithm Arrival Time Calculation

Given *partial*, k , *leaveTime* and jobSent
if (k -1) is divisible by m **and** ((k -1) / m) > 0
Increase *jobSent* by $m \cdot K$
Set *numJobs* $\coloneqq$ (floor (k / m)) $\cdot K$ + min(K , n -*jobSent*)
Set *sumProc1* to sum of processing times for the first *numJobs* jobs

Set *leaveTime* := max(*leaveTime* + 2*md* , *sumProc1*)
end if
Set *arriveTime*[*k*] := *leaveTime* + *travelTime*[*k*]
Return *arriveTime*[*k*]

A.3 Lower Bound 5

Algorithm of Lower Bound 5

Sort jobs and plants and reindex // d_i is the distance of plant with index i
Initialize array *A* //this array holds the values named as A_i in Table 4.4
Initialize array *noJobs* //this array holds the number of jobs on each trip
Set $LB := 0$, $jobSent := 0$, $trip := 0$
Set d to closest plant distance
Set *sumDist* to the sum of all plant distances
for each plant Pl_i
    Set $A[i] := 0$
end for
Set $A[1] := d$
//calculate the A_i values given in Table 4.4
for each plant Pl_i except the first plant
  for j from m to $(m+2-i)$ (decrement by 1)
    Increase $A[i]$ by $2d_j$
  end for
  Increase $A[i]$ by d_{m-i+1}
end for
while $jobSent < n$
  Increment *trip*
  if $trip \le m$
    Set $Sum := 0$
    Set $l_k := 0$ // This was called as l_k in Table 4.4
    //calculate l_k value
    for each job j
      if $Sum + p_j \le A[trip]$

Increase *Sum* by p_j and increment l_k

else break

end if

end for

Set *noJobs*[*trip*] := min(*n* - *jobSent*, min(K , l_k))

Increase *jobSent* by *noJobs*[*trip*]

else

Set *Sum* := 0

Set l_k := 0

// Determine which trip is the current trip of a round (i.e. 1st, 2nd, 3rd, ..., m^{th}) and set *factor* to this value

Set *factor* := *trip* - $\left\lfloor \frac{trip-1}{m} \right\rfloor \cdot m$

Set *round* := $\left\lfloor \frac{trip-1}{m} \right\rfloor$ //Here if truck is on its second round, *round* will be 1

for each job j

if *Sum* + $p_j \le 2 \cdot tour \cdot sumDist + A[factor]$

Increase *Sum* by p_j and increment l_k

else break

end if

end for

Set *Sum2* := 0

Set $l_{\min}$:= 0

for each job j

if *Sum2* + $p_j \le (2 \cdot tour - 1) \cdot d$ **and** $l_{\min} < tour \cdot K$

Increase *Sum2* by p_j and increment $l_{\min}$

else break

end if

end for

Set *noJobs*[*trip*] := min(*n* - *jobSent*, min(l_k - $l_{\min}$, K)

Increase *jobSent* by *noJobs*[*trip*]

end if

end while

for each trip t_k // the number of trips is equal to the value of *trip*, hence $k \le trip$

Increase *LB* by $2 \cdot k \cdot d \cdot$ *noJobs*[*trip*]

end for

Return *LB*

Appendix B

Pseudocodes of the Proposed Heuristics

B.1 Heuristic 1 (LPH1)

Algorithm Heuristic 1 (LPH1)

Sort plants in nondecreasing distances and reindex
Sort jobs in nondecreasing processing times and reindex
Set $TCT := 0$ // *TCT* means total completion time
Set $minTrip := \left\lfloor \frac{n}{K} \right\rfloor$ // the number of trips that will be fully loaded
Set $partial := n - qK$ // the number of jobs in the partially loaded trip
Set $trip := 0$ // Denotes which trip is the current one, can be at most $\left\lceil \frac{n}{K} \right\rceil$
Initialize arrays *AT* and *LT* and set each element of these arrays to 0 // *AT* stores the time each plant becomes available for processing while *LT* stores the time truck leaves each plant

Initialize array *NP* and set each element to 0 // holds the number of jobs sent from each plant

Set $lastP := 0$ // Denotes the index of the plant visited on the previous trip
for the first *minTrip* number of trips
  Set $Sum := 0$
  for each job J_l such that $trip \cdot K + 1 \le l \le (trip + 1) \cdot K$
    Increase *Sum* by p_l // p_l is the processing time of job J_l
  end for
  if the number of previously assigned jobs of each plant is the same
    Increase $AT[1]$ by *Sum*
    Set $LT[1] := \max(AT[1], LT[lastP] + d_{lastP} + d_1)$
    Set $lastP := 1$
    Increment *trip*

Increase *NP* [1] by K

Increase *TCT* by $K \cdot \left(LT\left[lastP \right] + d_{lastP} \right)$

else // potential plant selection will be applied

Set *selP* $:= 0$

Determine *potential plants*

Set *selP* to the plant with smallest contribution

Increase *AT* [*selP*] by *Sum*

Set *LT* [*selP*] $:=$ max (*AT* [*selP*], *LT* [*lastP*] + d_{lastP} + d_{selP})

Set *lastP* $:=$ *selP*

Increment *trip*

Increase *NP* [*lastP*] by K

Increase *TCT* by $K \cdot \left(LT\left[lastP \right] + d_{lastP} \right)$

end if

end for

if *partial* > 0

Set *Sum* $:= 0$

for each job J_l such that $minTrip \cdot K + 1 \le l \le n$

Increase *Sum* by p_l // p_l is the processing time of job J_l

end for

if the number of previously assigned jobs of each plant is the same

Increase *AT* [1] by *Sum*

Set *LT* [1] $:=$ max (*AT* [1], *LT* [*lastP*] + d_{lastP} + d_1)

Set *lastP* $:= 1$

Increment *trip*

Increase *NP* [1] by *partial*

Increase *TCT* by $partial \cdot \left(LT\left[lastP \right] + d_{lastP} \right)$

else // potential plant selection will be applied

Set *selP* $:= 0$

Determine *potential plants*

Set *selP* to the plant with smallest contribution

Increase *AT* [*selP*] by *Sum*

Set *LT* [*selP*] $:=$ max (*AT* [*selP*], *LT* [*lastP*] + d_{lastP} + d_{selP})

Set *lastP* $:=$ *selP*

Increment *trip*

Increase *NP* [*lastP*] by *partial*

Increase *TCT* by $partial \cdot \left(LT\left[lastP \right] + d_{lastP} \right)$

end if

end if

Return *TCT*

Algorithm Potential Plant Determination

Set $selP := 0$ // Denotes the plant selected among potential plants
Set $ratio := NP[1] \div K$ // Since plant 1 will always be the plant with more number of jobs
Set $count := 0$
for r from 0 to *ratio*
Set $count2 := 0$ // with this counter we ensure that among the plants having the same number of previously assigned jobs, only the closest one is chosen

for each plant P_i // P_i denotes Plant i
if $NP[i] = r \cdot K$ **and** $count2 < 1$
Determine P_i as a potential plant
Increment *count* and *count2*
end if
end for
end for
for each plant P_i
if P_i is among the potential plants
Set contribution of P_i to max $(LT[lastP] + d_{lastP} + d_i\ , AT[i] + Sum) + d_i$
end if
end for
Sort the potential plants in non-decreasing order of contributions

B.2 Heuristic 2 (LPH2)

Algorithm Heuristic 2 (LPH2)

Sort plants in non-decreasing distances and re-index
Sort jobs in non-decreasing processing times and re-index
Set $TCT := 0$ // *TCT* means total completion time
Set $minTrip := \left\lfloor \frac{n}{K} \right\rfloor$ // the number of trips that will be fully loaded
Set $partial := n - qK$ // the number of jobs in the partially loaded trip
Set $trip := 0$ // Denotes which trip is the current one, can be at most $\left\lceil \frac{n}{K} \right\rceil$

Initialize arrays *AT* and *LT* and set each element of these arrays to 0 // *AT* stores the time each plant becomes available for processing while *LT* stores the time truck leaves each plant

Set $lastP := 0$ // Denotes the index of the plant visited on the previous trip
Set $initSum := 0$
for each job J_l such that $1 \le l \le K$
    Increase *initSum* by p_l // p_l is the processing time of job J_l
end for
if $initSum > d_1$
    Increase *AT* [1] by *initSum*
    Set $LT[1] := initSum$
    Mark the first K jobs as assigned and sent
    Increase *TCT* by $K \cdot (LT[1] + d_1)$
    Increment *trip*
    Set $lastP := 1$
else
    Set $i := 1$
    Determine the *batch* to assign by *batch sliding*
    Set $sumK := 0$
    for each ij such that $1 \le ij \le K$
        Increase *sumK* by $p_{batch[ij]}$
        Mark job J_{ij} as assigned and sent
    end for
    Increase *AT* [1] by *sumK*
    Set $LT[1] := d_1$
    Increase *TCT* by $2 \cdot K \cdot d_1$
    Increment *trip*
    Set $lastP := 1$
end if
while $trip < minTrip$
    Set *batchEmpty* := true
    for each plant P_i **and** *batchEmpty* = true
        Set $assignP := i$
        Determine the *batch* by *batch sliding*
    end for
    if *batchEmpty* = false
        Set $sumK := 0$
        for each ij such that $1 \le ij \le K$
            Increase *sumK* by $p_{batch[ij]}$
            Mark job J_{ij} as assigned and sent

end for
Increase AT [*assignP*] by *sumK*
Set LT [*assignP*] := LT [*lastP*] + d_{lastP} + $d_{assignP}$
Increase *TCT* by $K \cdot \left(LT\left[assignP \right] + d_{assignP} \right)$
Increment *trip*
Set *lastP* := *assignP*
else
Set *Sum* to the sum of processing times of the first K unassigned jobs
Set *cont* to a very large value
for each plant P_i
Set contribution of P_i to (AT [i] + *Sum* + d_i)
if contribution of P_i ≤ *cont*
Set *selP* := i
end if
end for
Increase AT [*selP*] by *Sum*
Set LT [*selP*] := AT [*selP*]
Mark the first K jobs as assigned and sent
Increase *TCT* by $K \cdot \left(LT\left[selP \right] + d_{selP} \right)$
Increment *trip*
Set *lastP* := *selP*
end if
end while
if *partial* > 0
Set *Sum* to the sum of processing times of all unassigned jobs
Set *cont* to a very large value
for each plant P_i
Set contribution of P_i to max (LT [*lastP*] + d_{lastP} + d_i , AT [i] + *Sum*) + d_i
if contribution of P_i ≤ *cont*
Set *selP* := i
end if
end for
Increase AT [*selP*] by *Sum*
Set LT [*selP*] := max (LT [*lastP*] + d_{lastP} + d_{selP} , AT [*selP*])
Mark all unassigned jobs as assigned and sent
Increase *TCT* by $K \cdot \left(LT\left[selP \right] + d_{selP} \right)$
Increment *trip*
Set *lastP* := *selP*
end if
Return *TCT*

Algorithm Batch Sliding

for each job J_l such that $1 \le l \le n-K+1$
  Set *sum* to the sum of processing times of the first K unassigned jobs starting with l^{th} job
  if $sum \le LT[lastP] + d_{lastP} + d_i - AT[i]$
    Set *batchEmpty* := false
    for l' from 1 to K
      Set *batch* $[l']$ to the $(l')^{th}$ unassigned job starting with l^{th} job
    end for
  else break
  end if
end for
Return *batch*

B.3 Heuristic 3 (FTH1)

Algorithm Heuristic 3 (FTH1)

Sort plants in nondecreasing distances and reindex
Sort jobs in nondecreasing processing times and reindex
Set *TCT* := 0 // *TCT* means total completion time
Set *bestCT* to a very large value //holds the best total completion time
Set *minTrip* := $\left\lfloor \frac{n}{K} \right\rfloor$ // the number of trips that will be fully loaded
Set *partial* := $n - qK$ // the number of jobs in the partially loaded trip

if partial =0
  Apply Heuristic 1
  Set *bestCT* to the total completion time found in Heuristic 1
else //if a partial batch occurs
  for *p* from 0 to *minTrip* //*p* denotes which trip is partial, if the first trip is partial *p* will be 0
    Set *TCT* := 0

Set $trip := 0$ // Denotes which trip is the current one, can be at most $\left\lceil \frac{n}{K} \right\rceil$

Initialize arrays *AT* and *LT* and set each element of these arrays to 0 // *AT* stores the time each plant becomes available for processing while *LT* stores the time truck leaves each plant

Initialize array *NP* and set each element to 0 // holds the number of jobs sent from each plant

Set $lastP := 0$ // Denotes the index of the plant visited on the previous trip

if $p = 0$ // if the first trip is partial

Set *Sum* to the total processing time of the first *partial* jobs

Increase *AT* [1] by *Sum*

Set *LT* [1] := max (*AT* [1], *LT* [*lastP*] + $d_{lastP} + d_1$)

Set *lastP* := 1

Increment *trip*

Increase *NP* [1] by *partial*

Increase *TCT* by $partial \cdot \left(LT\left[lastP \right] + d_{lastP} \right)$

for the last *minTrip* number of trips

Set $Sum := 0$

for each job J_l such that $partial + 1 + (trip - 1)K \le l \le partial + trip \cdot K$

Increase *Sum* by p_l // p_l is the processing time of job J_l

end for

if the number of previously assigned jobs of each plant is the same

Increase *AT* [1] by *Sum*

Set *LT* [1] := max (*AT* [1], *LT* [*lastP*] + $d_{lastP} + d_1$)

Set *lastP* := 1

Increment *trip*

Increase *NP* [1] by *K*

Increase *TCT* by $K \cdot \left(LT\left[lastP \right] + d_{lastP} \right)$

else // potential plant selection will be applied

Set $selP := 0$

Determine *potential plants*

Set *selP* to the plant with smallest contribution

Increase *AT* [*selP*] by *Sum*

Set *LT* [*selP*] := max (*AT* [*selP*], *LT* [*lastP*] + $d_{lastP} + d_{selP}$)

Set $lastP := selP$

Increment *trip*

Increase *NP* [*lastP*] by *K*

Increase *TCT* by $K \cdot \left(LT\left[lastP \right] + d_{lastP} \right)$

end if

end for

if $TCT \leq bestCT$
Set $bestCT := TCT$
end if
else // a trip other than the first one is partial
for each *trip*
Set $Sum := 0$
if $trip < p$
for each job J_l such that $trip \cdot K + 1 \leq l \leq (trip + 1)K$
Increase *Sum* by p_l // p_l is the processing time of job J_l
end for
else if $trip = p$
for each job J_l such that $p \cdot K + 1 \leq l \leq p \cdot K + partial$
Increase *Sum* by p_l // p_l is the processing time of job J_l
end for
else
for each job J_l such that $(trip - 1) \cdot K + partial + 1 \leq l \leq trip \cdot K + partial$
Increase *Sum* by p_l // p_l is the processing time of job J_l
end for
end if
if the number of previously assigned jobs of each plant is the same
Increase $AT[1]$ by *Sum*
Set $LT[1] := \max(AT[1], LT[lastP] + d_{lastP} + d_1)$
Set $lastP := 1$
Increment *trip*
if $trip - 1 \neq p$
Increase $NP[1]$ by K
Increase *TCT* by $K \cdot \left(LT[lastP] + d_{lastP}\right)$
else
Increase $NP[1]$ by *partial*
Increase *TCT* by $partial \cdot \left(LT[lastP] + d_{lastP}\right)$
end if
else
Set $selP := 0$
Determine *potential plants*
Set *selP* to the plant with smallest contribution
Increase $AT[selP]$ by *Sum*
Set $LT[selP] := \max(AT[selP], LT[lastP] + d_{lastP} + d_{selP})$
Set $lastP := selP$
Increment *trip*
if $trip - 1 \neq p$

Increase *NP* [*lastP*] by *K*
Increase *TCT* by $K \cdot \left(LT\left[lastP \right] + d_{lastP} \right)$
else
Increase *NP* [*lastP*] by *partial*
Increase *TCT* by $partial \cdot \left(LT\left[lastP \right] + d_{lastP} \right)$
end if
end if
end for
if $TCT \leq bestCT$
Set $bestCT := TCT$
end if
end if
end for
end if
Return *bestCT*

Algorithm Potential Plant Determination 2

Set $selP := 0$ // Denotes the plant selected among potential plants
Set $maxSent := 0$ // Since plant 1 will always be the plant with more number of jobs
for each plant P_i
if $NP[i] \geq maxSent$
Set $maxSent := NP[i]$
end if
end for

Set $count := 0$
for *r* from 0 to *maxSent*
Set $count2 := 0$ // with this counter we ensure that among the plants having the same number of previously assigned jobs, only the closest one is chosen
for each plant P_i // P_i denotes Plant i
if *NP* [*i*] = *r* **and** *count2* < 1
Determine P_i as a potential plant
Increment *count* and *count2*
end if
end for
end for
for each plant P_i
if P_i is among the potential plants
Set contribution of P_i to max (LT [*lastP*] + d_{lastP} + d_i , AT [*i*] + *Sum*) + d_i

end if
end for
Sort the potential plants in non-decreasing order of contributions

B.4 Heuristic 4 (FTH2)

Algorithm Heuristic 4 (FTH2)

Sort plants in non-decreasing distances and re-index
Sort jobs in non-decreasing processing times and re-index
Set $TCT := 0$ // *TCT* means total completion time
Set *bestCT* to a very large value //holds the best total completion time
Set $minTrip := \left\lfloor \frac{n}{K} \right\rfloor$ // the number of trips that will be fully loaded
Set $partial := n - qK$ // the number of jobs in the partially loaded trip
Set $trip := 0$ // Denotes which trip is the current one, can be at most $\left\lceil \frac{n}{K} \right\rceil$
if $partial = 0$
Apply Heuristic 2
Set *bestCT* to the total completion time found in Heuristic 2
else
for *p* from 1 to *minTrip*+1 // *p* denotes which trip is partial
Set $TCT := 0$
Set $trip := 0$ // Denotes which trip is the current one, can be at most $\left\lceil \frac{n}{K} \right\rceil$
Initialize arrays *AT* and *LT* and set each element of these arrays to 0 // *AT* stores the time each plant becomes available for processing while *LT* stores the time truck leaves each plant
Initialize array *NP* and set each element to 0 // holds the number of jobs sent from each plant
Set $lastP := 0$ // Denotes the index of the plant visited on the previous trip
Set $batchEmpty := \text{true}$
if $p = 1$ // if the first trip is partial
Set *initSum* to the total processing time of the first *partial* jobs
if $initSum > d_1$

Increase *AT* [1] by *initSum*
Set *LT* [1] := *initSum*
Mark the first *partial* jobs as assigned and sent
Increase *TCT* by $partial \cdot \left(LT\left[1\right] + d_1 \right)$
Increment *trip*
Set *lastP* := 1
else
Set $i := 1$
Determine the *batch* to assign by *batch sliding*
Set *sumK* := 0
for each ij such that $1 \leq ij \leq partial$
Increase *sumK* by $p_{batch[ij]}$
Mark job J_{ij} as assigned and sent
end for
Increase *AT* [1] by *sumK*
Set *LT* [1] := d_1
Increase *TCT* by $2 \cdot partial \cdot d_1$
Increment *trip*
Set *lastP* := 1
end if
while *trip* < *minTrip*+1
Set *batchEmpty* := true
for each plant P_i **and** *batchEmpty* = true
Set *assignP* := i
Determine the *batch* by *batch sliding*
end for
if *batchEmpty* = false
Set *sumK* := 0
for each ij such that $1 \leq ij \leq K$
Increase *sumK* by $p_{batch[ij]}$
Mark job J_{ij} as assigned and sent
end for
Increase *AT* [*assignP*] by *sumK*
Set *LT* [*assignP*] := *LT* [*lastP*] + d_{lastP} + $d_{assignP}$
Increase *TCT* by $K \cdot \left(LT\left[assignP\right] + d_{assignP} \right)$
Increment *trip*
Set *lastP* := *assignP*
else
Set *Sum* to the sum of processing times of the first K unassigned jobs
Set *cont* to a very large value

for each plant P_i

Set contribution of P_i to $(AT[i] + Sum + d_i)$

if contribution of $P_i \leq cont$

Set $selP := i$

end if

end for

Increase $AT[selP]$ by Sum

Set $LT[selP] := AT[selP]$

Mark the first K jobs as assigned and sent

Increase TCT by $K \cdot \left(LT\left[selP\right] + d_{selP}\right)$

Increment *trip*

Set $lastP := selP$

end if

end while

if $TCT \leq bestCT$

Set $bestCT := TCT$

end if

else // if a trip other than the first one is partial

Set *initSum* to the total processing time of the first K jobs

if $initSum > d_1$

Increase $AT[1]$ by *initSum*

Set $LT[1] := initSum$

Mark the first *partial* jobs as assigned and sent

Increase TCT by $K \cdot \left(LT\left[1\right] + d_1\right)$

Increment *trip*

Set $lastP := 1$

else

Set $i := 1$

Determine the *batch* to assign by *batch sliding*

Set $sumK := 0$

for each ij such that $1 \leq ij \leq K$

Increase *sumK* by $p_{batch[ij]}$

Mark job J_{ij} as assigned and sent

end for

Increase $AT[1]$ by *sumK*

Set $LT[1] := d_1$

Increase TCT by $2 \cdot K \cdot d_1$

Increment *trip*

Set $lastP := 1$

end if

while *trip* +1<*p*
Set *batchEmpty* := true
for each plant P_i **and** *batchEmpty* = true
Set *assignP* := i
Determine the *batch* by *batch sliding*
end for
if *batchEmpty* = false
Set *sumK* := 0
for each ij such that $1 \le ij \le K$
Increase *sumK* by $p_{batch[ij]}$
Mark job J_{ij} as assigned and sent
end for
Increase *AT* [*assignP*] by *sumK*
Set *LT* [*assignP*] := *LT* [*lastP*] + d_{lastP} + $d_{assignP}$
Increase *TCT* by $K \cdot \left(LT\left[assignP \right] + d_{assignP} \right)$
Increment *trip*
Set *lastP* := *assignP*
else
Set *Sum* to the sum of processing times of the first K unassigned jobs
Set *cont* to a very large value
for each plant P_i
Set contribution of P_i to (*AT* [i] + *Sum* + d_i)
if contribution of $P_i \le cont$
Set *selP* := i
end if
end for
Increase *AT* [*selP*] by *Sum*
Set *LT* [*selP*] := *AT* [*selP*]
Mark the first K jobs as assigned and sent
Increase *TCT* by $K \cdot \left(LT\left[selP \right] + d_{selP} \right)$
Increment *trip*
Set *lastP* := *selP*
end if
end while
if *trip*+1=*p*
if *trip*+1=*minTrip*+1 //if the last trip is partial
Set *Sum* to the sum of processing times of all unassigned jobs
Set *cont* to a very large value
for each plant P_i
Set contribution of P_i to max (*LT* [*lastP*] + d_{lastP} + d_i , *AT* [i] + *Sum*) + d_i

if contribution of $P_i \leq cont$

Set $selP := i$

end if

end for

Increase AT [$selP$] by Sum

Set LT [$selP$] := max (LT [$lastP$] + d_{lastP} + d_{selP} , AT [$selP$])

Mark all unassigned jobs as assigned and sent

Increase TCT by $K \cdot \left(LT\left[selP \right] + d_{selP} \right)$

Increment *trip*

Set $lastP := selP$

else

Set *batchEmpty* := true

for each plant P_i **and** *batchEmpty* = true

Set $assignP := i$

Determine the *batch* by *batch sliding*

end for

if *batchEmpty* = false

Set $sumK := 0$

for each ij such that $1 \leq ij \leq K$

Increase *sumK* by $p_{batch[ij]}$

Mark job J_{ij} as assigned and sent

end for

Increase AT [$assignP$] by $sumK$

Set LT [$assignP$] := LT [$lastP$] + d_{lastP} + $d_{assignP}$

Increase TCT by $K \cdot \left(LT\left[assignP \right] + d_{assignP} \right)$

Increment *trip*

Set $lastP := assignP$

else

Set *Sum* to the sum of processing times of the first K unassigned jobs

Set *cont* to a very large value

for each plant P_i

Set contribution of P_i to (AT [i] + Sum + d_i)

if contribution of $P_i \leq cont$

Set $selP := i$

end if

end for

Increase AT [$selP$] by Sum

Set LT [$selP$] := AT [$selP$]

Mark the first K jobs as assigned and sent

Increase *TCT* by $K \cdot \left(LT\left[selP \right] + d_{selP} \right)$

Increment *trip*

Set *lastP* := *selP*

end if

end if

end if

if $p \neq minTrip+1$

while *trip* < *minTrip*+1

Set *batchEmpty* := true

for each plant P_i **and** *batchEmpty* = true

Set *assignP* := i

Determine the *batch* by *batch sliding*

end for

if *batchEmpty* = false

Set *sumK* := 0

for each ij such that $1 \leq ij \leq K$

Increase *sumK* by $p_{batch[ij]}$

Mark job J_{ij} as assigned and sent

end for

Increase *AT* [*assignP*] by *sumK*

Set *LT* [*assignP*] := *LT* [*lastP*] + d_{lastP} + $d_{assignP}$

Increase *TCT* by $K \cdot \left(LT\left[assignP \right] + d_{assignP} \right)$

Increment *trip*

Set *lastP* := *assignP*

else

Set *Sum* to the sum of processing times of the first K unassigned jobs

Set *cont* to a very large value

for each plant P_i

Set contribution of P_i to (*AT* [i] + *Sum* + d_i)

if contribution of $P_i \leq cont$

Set *selP* := i

end if

end for

Increase *AT* [*selP*] by *Sum*

Set *LT* [*selP*] := *AT* [*selP*]

Mark the first K jobs as assigned and sent

Increase *TCT* by $K \cdot \left(LT\left[selP \right] + d_{selP} \right)$

Increment *trip*

Set *lastP* := *selP*

```
            end if
        end while
    end if
    if TCT ≤ bestCT
        Set bestCT := TCT
    end if
  end if
 end for
end if
Return bestCT
```

B.5 Heuristic 5 (NWH1)

Algorithm Heuristic 5 (NWH1)

```
Sort plants in nondecreasing distances and reindex
Sort jobs in nondecreasing processing times and reindex
Set TCT := 0      // TCT means total completion time
Set numSent := 0, lastP := 0
Initialize arrays AT and LT and set each element of these arrays to 0      // AT stores the time
each plant becomes available for processing while LT stores the time truck leaves each plant

Initialize array NP and set each element to 0      // holds the number of jobs sent from each
plant
Set ptotal to the sum of processing times of all jobs
Set pbar := ptotal ÷ n
while numSent < n
   for each plant P_i
      Set NP [i] := 0
   end for
   Set sum := 0
   for each job J_l such that numSent + 1 ≤ l ≤ n
      if AT [1] + sum + p_l ≤ LT [lastP] + d_lastP + d_1 and NP [1] < K
         Increase sum by p_l
                  Increment NP [1]
      else break
      end if
```

end for
if $NP\,[1] = K$
Increase *numSent* by $NP\,[1]$
Increase $AT\,[1]$ by *sum*
Set $LT\,[1] := LT\,[lastP] + d_{lastP} + d_1$
Set $lastP := 1$
Increase *TCT* by $K \cdot \left(LT\left[1\right] + d_1\right)$
else
Set *justified* := false
for each plant P_i except the first plant
Set $NP\,[i] := 0$
Set $sum := 0$
for each job J_l such that $numSent + 1 \le l \le n$
if $AT\,[i] + sum + p_l \le LT\,[lastP] + d_{lastP} + d_i$ **and** $NP\,[i] < K$
Increase *sum* by p_l
Increment $NP\,[i]$
else break
end if
end for
if $2 \cdot (d_i - d_1) / pbar < (NP\,[i] - NP\,[1])$ **and** $(NP\,[i] - NP\,[1]) \ge 1$
Increase *numSent* by $NP\,[i]$
Increase $AT\,[i]$ by *sum*
Set $LT\,[i] := LT\,[lastP] + d_{lastP} + d_i$
Set $lastP := i$
Increase *TCT* by $NP\left[i\right] \cdot \left(LT\left[i\right] + d_i\right)$
Set *justified* = true
break
end if
end for
if *justified* = false
Increase *numSent* by $NP\,[1]$
Increase $AT\,[1]$ by *sum*
Set $LT\,[1] := LT\,[lastP] + d_{lastP} + d_1$
Set $lastP := 1$
Increase *TCT* by $NP\left[1\right] \cdot \left(LT\left[1\right] + d_1\right)$
end if
end if
end while
Return *TCT*

B.6 Heuristic 6 (NWH2)

Algorithm Heuristic 6 (NWH2)

Sort plants in nondecreasing distances and reindex
Sort jobs in nondecreasing processing times and reindex
Set $TCT := 0$ // *TCT* means total completion time
Set $numSent := 0$, $lastP := 0$
Initialize arrays *AT* and *LT* and set each element of these arrays to 0 // *AT* stores the time each plant becomes available for processing while *LT* stores the time truck leaves each plant

Initialize array *NP* and set each element to 0 // holds the number of jobs sent from each plant

Set *ptotal* to the sum of processing times of all jobs
Set $pbar := ptotal \div n$
while $numSent < n$
  for each plant P_i
    Set $NP[i] := 0$
  end for
  Set $sum := 0$
  for each job J_l such that $numSent + 1 \le l \le n$
    if $AT[1] + sum + p_l \le LT[lastP] + d_{lastP} + d_1$ **and** $NP[1] < K$
      Increase *sum* by p_l
        Increment $NP[1]$
    else break
    end if
  end for
  if $NP[1] = K$
    Set $sum2 := 0$
    Apply *batch sliding*
    for each ij such that $1 \le ij \le NP[1]$
      Increase *sum2* by $p_{batch[ij]}$
      Mark job J_{ij} as assigned and sent
    end for
    Increase *numSent* by $NP[1]$
    Increase $AT[1]$ by *sum2*
    Set $LT[1] := LT[lastP] + d_{lastP} + d_1$
    Set $lastP := 1$

Increase *TCT* by $K \cdot \left(LT[1] + d_1 \right)$

else

Set *justified* := false

for each plant P_i except the first plant

Set *NP* [*i*] := 0

Set *sum* := 0

for each job J_l such that $numSent + 1 \le l \le n$

if $AT[i] + sum + p_l \le LT[lastP] + d_{lastP} + d_i$ **and** $NP[i] < K$

Increase *sum* by p_l

Increment *NP* [*i*]

else break

end if

end for

if $2 \cdot (d_i - d_1) / pbar <$ (*NP* [*i*] - *NP* [1]) **and** (*NP* [*i*] - *NP* [1]) ≥ 1

Set *sum2* := 0

Apply *batch sliding*

for each *ij* such that $1 \le ij \le NP[i]$

Increase *sum2* by $p_{batch[ij]}$

Mark job J_{ij} as assigned and sent

end for

Increase *numSent* by *NP* [*i*]

Increase *AT* [*i*] by *sum2*

Set $LT[i] := LT[lastP] + d_{lastP} + d_i$

Set *lastP* := *i*

Increase *TCT* by $NP[i] \cdot \left(LT[i] + d_i \right)$

Set *justified* = true

break

end if

end for

if *justified* = false

Set *sum2* := 0

Apply *batch sliding*

for each *ij* such that $1 \le ij \le NP[1]$

Increase *sum2* by $p_{batch[ij]}$

Mark job J_{ij} as assigned and sent

end for

Increase *numSent* by *NP* [1]

Increase *AT* [1] by *sum2*

Set $LT[1] := LT[lastP] + d_{lastP} + d_1$

Set *lastP* := 1
Increase *TCT* by $NP[1] \cdot (LT[1] + d_1)$
end if
end if
end while
Return *TCT*

B.7 Heuristic 7 (GH)

Algorithm Heuristic 7 (GH)

Sort plants in nondecreasing distances and reindex
Sort jobs in nondecreasing processing times and reindex
Initialize array *NP* and set each element to 0 // holds the number of jobs sent from each plant
Set *bestCT* to a very large value // Denotes the best completion time
Set *ptotal* to the sum of processing times of all jobs
Set *pbar* := *ptotal* ÷ *n*
for each *ratio-to-wait* such that $0.1 \leq$ *ratio-to-wait* ≤ 1 (Increase by 0.1)
Initialize arrays *AT* and *LT* and set each element of these arrays to 0 // *AT* stores the time each plant becomes available for processing while *LT* stores the time truck leaves each plant

Set *TCT* := 0 // *TCT* means total completion time
Set *numSent* := 0, *lastP* := 0
while *numSent* < *n*
Set *batchEmpty* := true
for each plant P_i
Set *NP* [*i*] := 0
end for
Set *sum* := 0
for each job J_l such that $1 \leq l \leq n$
if *AT* [1] + *sum* + p_l ≤ *LT* [*lastP*] + d_{lastP} + d_1 **and** *NP* [1] < *K* **and** J_l is unassigned
Increase *sum* by p_l
Increment *NP* [1]
else

if $AT\,[1] + sum + (ratio\text{-}to\text{-}wait)\cdot p_l \le LT\,[lastP] + d_{lastP} + d_1$ **and** $NP\,[1] < K$ **and** J_l is unassigned

Increase *sum* by p_l

Increment $NP\,[1]$

end if

end if

end for

if $NP\,[1] = K$

Apply *batch sliding*

if *batchEmpty* = false

Set $sum2 := 0$

for each ij such that $1 \le ij \le NP\,[1]$

Increase *sum2* by $p_{batch[ij]}$

Mark job J_{ij} as assigned and sent

end for

Increase *numSent* by $NP\,[1]$

Increase $AT\,[1]$ by *sum2*

Set $LT\,[1] := \max(AT\,[1], LT\,[lastP] + d_{lastP} + d_1)$

Set $lastP := 1$

Increase *TCT* by $K\cdot\left(LT\left[1\right]+d_1\right)$

else

Set *sum2* to the sum of processing times of the first $NP\,[1]$ unassigned jobs

Mark the first $NP\,[1]$ jobs as assigned and sent

Increase $AT\,[1]$ by *sum2*

Set $LT\,[1] := \max(AT\,[1], LT\,[lastP] + d_{lastP} + d_1)$

Set $lastP := 1$

Increase *TCT* by $K\cdot\left(LT\left[1\right]+d_1\right)$

end if

else

Set *justified* := false

for each plant P_i except the first plant

Set $NP\,[i] := 0$

Set $sum := 0$

for each job J_l such that $numSent + 1 \le l \le n$

if $AT\,[i] + sum + p_l \le LT\,[lastP] + d_{lastP} + d_i$ **and** $NP\,[i] < K$ **and** J_l is unassigned

Increase *sum* by p_l

Increment $NP\,[i]$

else

if $AT[i] + sum + (ratio\text{-}to\text{-}wait) \cdot p_l \leq LT[lastP] + d_{lastP} + d_i$ **and** $NP[i] < K$ **and** J_l is unassigned

Increase *sum* by p_l

Increment $NP[i]$

end if

end if

end for

if $2 \cdot (d_i - d_1) / pbar < (NP[i] - NP[1])$ **and** $(NP[i] - NP[1]) \geq 1$

Apply *batch sliding*

if *batchEmpty* = false

Set $sum2 := 0$

for each ij such that $1 \leq ij \leq NP[i]$

Increase *sum2* by $p_{batch[ij]}$

Mark job J_{ij} as assigned and sent

end for

Increase *numSent* by $NP[i]$

Increase $AT[i]$ by *sum2*

Set $LT[i] := \max(AT[i], LT[lastP] + d_{lastP} + d_i)$

Set $lastP := i$

Increase *TCT* by $NP[i] \cdot (LT[i] + d_i)$

Set *justified* = true

break

else

Set *sum2* to the sum of processing times of the first $NP[i]$ unassigned jobs

Mark the first $NP[i]$ jobs as assigned and sent

Increase $AT[i]$ by *sum2*

Set $LT[i] := \max(AT[i], LT[lastP] + d_{lastP} + d_i)$

Set $lastP := i$

Increase *TCT* by $NP[i] \cdot (LT[i] + d_i)$

Set *justified* = true

break

end if

end if

end for

if *justified* = false

Apply *batch sliding*

if *batchEmpty* = false

Set $sum2 := 0$

for each ij such that $1 \leq ij \leq NP[1]$

Increase *sum2* by $p_{batch[ij]}$

Mark job J_{ij} as assigned and sent

end for

Increase *numSent* by *NP* [1]

Increase *AT* [1] by *sum2*

Set *LT* [1] := max (*AT* [1], *LT* [*lastP*] + d_{lastP} + d_1)

Set *lastP* := 1

Increase *TCT* by $NP[1]\cdot\left(LT[1]+d_1\right)$

else

Set *sum2* to the sum of processing times of the first *NP* [1] unassigned jobs

Mark the first *NP* [1] jobs as assigned and sent

Increase *AT* [1] by *sum2*

Set *LT* [1] := max (*AT* [1], *LT* [*lastP*] + d_{lastP} + d_1)

Set *lastP* := 1

Increase *TCT* by $NP[1]\cdot\left(LT[1]+d_1\right)$

end if

end if

end if

end while

if *TCT* ≤ *bestCT*

Set *bestCT* := *TCT*

end if

end for

Return *bestCT*

Appendix C

Numerical Results for Last Trip Partial Problem

	K	Ins. #	*LastPartialHR1* (LPHR1)	*LastPartialHR2* (LPHR2)	Lower Bound	Lower Bound Gap (%)		Average Lower Bound Gap (%)		Maximum Lower Bound Gap (%)	
						LPHR1	LPHR2	LPHR1	LPHR2	LPHR1	LPHR2
10	**2**	1	1560	1560	1560	0.00	0.00	0.00	0.00	0.00	0.00
		2	1440	1440	1440	0.00	0.00				
		3	1380	1380	1380	0.00	0.00				
		4	1560	1560	1560	0.00	0.00				
		5	1560	1560	1560	0.00	0.00				
10	**4**	1	936	936	936	0.00	0.00	0.00	0.00	0.00	0.00
		2	864	864	864	0.00	0.00				
		3	828	828	828	0.00	0.00				
		4	936	936	936	0.00	0.00				
		5	936	936	936	0.00	0.00				
10	**6**	1	728	728	728	0.00	0.00	0.00	0.00	0.00	0.00
		2	672	672	672	0.00	0.00				
		3	644	644	644	0.00	0.00				
		4	728	728	728	0.00	0.00				
		5	748	748	748	0.00	0.00				
10	**8**	1	724	724	724	0.00	0.00	0.00	0.00	0.00	0.00
		2	776	776	776	0.00	0.00				
		3	672	672	672	0.00	0.00				
		4	744	744	744	0.00	0.00				
		5	814	814	814	0.00	0.00				

Table C.1 : Computational results for *Last Trip Partial Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	*LastPartialHR1* (LPHR1)	*LastPartialHR2* (LPHR2)	Lower Bound	Lower Bound Gap (%)		Average Lower Bound Gap (%)		Maximum Lower Bound Gap (%)	
						LPHR1	LPHR2	LPHR1	LPHR2	LPHR1	LPHR2
25	**2**	1	8788	8788	8788	0.00	0.00	0.00	0.00	0.00	0.00
		2	8112	8112	8112	0.00	0.00				
		3	7774	7774	7774	0.00	0.00				
		4	8788	8788	8788	0.00	0.00				
		5	8788	8788	8788	0.00	0.00				
25	**4**	1	4732	4732	4732	0.00	0.00	0.00	0.00	0.00	0.00
		2	4368	4368	4368	0.00	0.00				
		3	4186	4186	4186	0.00	0.00				
		4	4732	4732	4732	0.00	0.00				
		5	4732	4732	4732	0.00	0.00				
25	**6**	1	3380	3380	3380	0.00	0.00	0.09	0.09	0.45	0.45
		2	3134	3120	3120	0.45	0.45				
		3	2990	2990	2990	0.00	0.00				
		4	3380	3380	3380	0.00	0.00				
		5	3380	3380	3380	0.00	0.00				
25	**8**	1	2722	2704	2704	0.67	0.67	0.68	0.68	1.31	1.31
		2	2630	2596	2596	1.31	1.31				
		3	2410	2392	2392	0.75	0.75				
		4	2722	2704	2704	0.67	0.67				
		5	2704	2704	2704	0.00	0.00				

Table C.1 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	*LastPartialHR1* (LPHR1)	*LastPartialHR2* (LPHR2)	Lower Bound	Lower Bound Gap (%)		Average Lower Bound Gap (%)		Maximum Lower Bound Gap (%)	
						LPHR1	LPHR2	LPHR1	LPHR2	LPHR1	LPHR2
40	**2**	1	21840	21840	21840	0.00	0.00	0.00	0.00	0.00	0.00
		2	20160	20160	20160	0.00	0.00				
		3	19320	19320	19320	0.00	0.00				
		4	21840	21840	21840	0.00	0.00				
		5	21840	21840	21840	0.00	0.00				
40	**4**	1	11440	11440	11440	0.00	0.00	0.00	0.00	0.00	0.00
		2	10560	10560	10560	0.00	0.00				
		3	10120	10120	10120	0.00	0.00				
		4	11440	11440	11440	0.00	0.00				
		5	11440	11440	11440	0.00	0.00				
40	**6**	1	8008	8008	8008	0.00	0.00	0.00	0.00	0.00	0.00
		2	7392	7392	7392	0.00	0.00				
		3	7084	7084	7084	0.00	0.00				
		4	8008	8008	8008	0.00	0.00				
		5	8008	8008	8008	0.00	0.00				
40	**8**	1	6256	6256	6240	0.26	0.26	0.33	0.33	0.58	0.58
		2	5792	5792	5760	0.56	0.56				
		3	5552	5552	5520	0.58	0.58				
		4	6256	6256	6240	0.26	0.26				
		5	6240	6240	6240	0.00	0.00				

Table C.1 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	*LastPartialHR1* (LPHR1)	*LastPartialHR2* (LPHR2)	Lower Bound	Lower Bound Gap (%)		Average Lower Bound Gap (%)		Maximum Lower Bound Gap (%)	
						LPHR1	LPHR2	LPHR1	LPHR2	LPHR1	LPHR2
55	**2**	1	40768	40768	40768	0.00	0.00	0.00	0.00	0.00	0.00
		2	37632	37632	37632	0.00	0.00				
		3	36064	36064	36064	0.00	0.00				
		4	40768	40768	40768	0.00	0.00				
		5	40768	40768	40768	0.00	0.00				
55	**4**	1	21112	21112	21112	0.00	0.00	0.00	0.00	0.00	0.00
		2	19488	19488	19488	0.00	0.00				
		3	18676	18676	18676	0.00	0.00				
		4	21112	21112	21112	0.00	0.00				
		5	21112	21112	21112	0.00	0.00				
55	**6**	1	14560	14560	14560	0.00	0.00	0.00	0.00	0.00	0.00
		2	13440	13440	13440	0.00	0.00				
		3	12880	12880	12880	0.00	0.00				
		4	14560	14560	14560	0.00	0.00				
		5	14560	14560	14560	0.00	0.00				
55	**8**	1	11298	11298	11284	0.12	0.12	0.14	0.14	0.29	0.29
		2	10446	10446	10416	0.29	0.29				
		3	9996	9996	9982	0.14	0.14				
		4	11298	11298	11284	0.12	0.12				
		5	11284	11284	11284	0.00	0.00				

Table C.1 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	*LastPartialHR1* (LPHR1)	*LastPartialHR2* (LPHR2)	Lower Bound	Lower Bound Gap (%)		Average Lower Bound Gap (%)		Maximum Lower Bound Gap (%)	
						LPHR1	LPHR2	LPHR1	LPHR2	LPHR1	LPHR2
10	**2**	1	2160	2160	2160	0.00	0.00	0.00	0.00	0.00	0.00
		2	2040	2040	2040	0.00	0.00				
		3	1980	1980	1980	0.00	0.00				
		4	2160	2160	2160	0.00	0.00				
		5	2160	2160	2160	0.00	0.00				
10	**4**	1	1296	1296	1296	0.00	0.00	0.00	0.00	0.00	0.00
		2	1224	1224	1224	0.00	0.00				
		3	1188	1188	1188	0.00	0.00				
		4	1296	1296	1296	0.00	0.00				
		5	1296	1296	1296	0.00	0.00				
10	**6**	1	1008	1008	1008	0.00	0.00	0.00	0.00	0.00	0.00
		2	952	952	952	0.00	0.00				
		3	924	924	924	0.00	0.00				
		4	1008	1008	1008	0.00	0.00				
		5	1008	1008	1008	0.00	0.00				
10	**8**	1	864	864	864	0.00	0.00	0.00	0.00	0.00	0.00
		2	916	916	916	0.00	0.00				
		3	812	812	812	0.00	0.00				
		4	884	884	884	0.00	0.00				
		5	954	954	954	0.00	0.00				

Table C.2 : Computational results for *Last Trip Partial Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	*LastPartialHR1* (LPHR1)	*LastPartialHR2* (LPHR2)	Lower Bound	Lower Bound Gap (%)		Average Lower Bound Gap (%)		Maximum Lower Bound Gap (%)	
						LPHR1	LPHR2	LPHR1	LPHR2	LPHR1	LPHR2
25	**2**	1	12168	12168	12168	0.00	0.00	0.00	0.00	0.00	0.00
		2	11492	11492	11492	0.00	0.00				
		3	11154	11154	11154	0.00	0.00				
		4	12168	12168	12168	0.00	0.00				
		5	12168	12168	12168	0.00	0.00				
25	**4**	1	6552	6552	6552	0.00	0.00	0.00	0.00	0.00	0.00
		2	6188	6188	6188	0.00	0.00				
		3	6006	6006	6006	0.00	0.00				
		4	6552	6552	6552	0.00	0.00				
		5	6552	6552	6552	0.00	0.00				
25	**6**	1	4680	4680	4680	0.00	0.00	0.00	0.00	0.00	0.00
		2	4420	4420	4420	0.00	0.00				
		3	4290	4290	4290	0.00	0.00				
		4	4680	4680	4680	0.00	0.00				
		5	4680	4680	4680	0.00	0.00				
25	**8**	1	3744	3744	3744	0.00	0.00	0.05	0.10	0.25	0.51
		2	3545	3554	3536	0.25	0.51				
		3	3432	3432	3432	0.00	0.00				
		4	3744	3744	3744	0.00	0.00				
		5	3744	3744	3744	0.00	0.00				

Table C.2 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	*LastPartialHR1* (LPHR1)	*LastPartialHR2* (LPHR2)	Lower Bound	Lower Bound Gap (%)		Average Lower Bound Gap (%)		Maximum Lower Bound Gap (%)	
						LPHR1	LPHR2	LPHR1	LPHR2	LPHR1	LPHR2
40	**2**	1	30240	30240	30240	0.00	0.00	0.00	0.00	0.00	0.00
		2	28560	28560	28560	0.00	0.00				
		3	27720	27720	27720	0.00	0.00				
		4	30240	30240	30240	0.00	0.00				
		5	30240	30240	30240	0.00	0.00				
40	**4**	1	15840	15840	15840	0.00	0.00	0.00	0.00	0.00	0.00
		2	14960	14960	14960	0.00	0.00				
		3	14520	14520	14520	0.00	0.00				
		4	15840	15840	15840	0.00	0.00				
		5	15840	15840	15840	0.00	0.00				
40	**6**	1	11088	11088	11088	0.00	0.00	0.00	0.00	0.00	0.00
		2	10472	10472	10472	0.00	0.00				
		3	10164	10164	10164	0.00	0.00				
		4	11088	11088	11088	0.00	0.00				
		5	11088	11088	11088	0.00	0.00				
40	**8**	1	8640	8640	8640	0.00	0.00	0.00	0.00	0.00	0.00
		2	8160	8160	8160	0.00	0.00				
		3	7920	7920	7920	0.00	0.00				
		4	8640	8640	8640	0.00	0.00				
		5	8640	8640	8640	0.00	0.00				

Table C.2 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	*LastPartialHR1* (LPHR1)	*LastPartialHR2* (LPHR2)	Lower Bound	Lower Bound Gap (%)		Average Lower Bound Gap (%)		Maximum Lower Bound Gap (%)	
						LPHR1	LPHR2	LPHR1	LPHR2	LPHR1	LPHR2
55	**2**	1	56448	56448	56448	0.00	0.00	0.00	0.00	0.00	0.00
		2	53312	53312	53312	0.00	0.00				
		3	51744	51744	51744	0.00	0.00				
		4	56448	56448	56448	0.00	0.00				
		5	56448	56448	56448	0.00	0.00				
55	**4**	1	29232	29232	29232	0.00	0.00	0.00	0.00	0.00	0.00
		2	27608	27608	27608	0.00	0.00				
		3	26796	26796	26796	0.00	0.00				
		4	29232	29232	29232	0.00	0.00				
		5	29232	29232	29232	0.00	0.00				
55	**6**	1	20160	20160	20160	0.00	0.00	0.00	0.00	0.00	0.00
		2	19040	19040	19040	0.00	0.00				
		3	18480	18480	18480	0.00	0.00				
		4	20160	20160	20160	0.00	0.00				
		5	20160	20160	20160	0.00	0.00				
55	**8**	1	15624	15624	15624	0.00	0.00	0.00	0.00	0.00	0.00
		2	14756	14756	14756	0.00	0.00				
		3	14322	14322	14322	0.00	0.00				
		4	15624	15624	15624	0.00	0.00				
		5	15624	15624	15624	0.00	0.00				

Table C.2 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	*LastPartialHR1* (LPHR1)	*LastPartialHR2* (LPHR2)	Lower Bound	Lower Bound Gap (%)		Average Lower Bound Gap (%)		Maximum Lower Bound Gap (%)	
						LPHR1	LPHR2	LPHR1	LPHR2	LPHR1	LPHR2
10	**2**	1	2400	2400	2400	0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860	0.00	0.00				
		3	1680	1680	1680	0.00	0.00				
		4	2220	2220	2220	0.00	0.00				
		5	2340	2340	2340	0.00	0.00				
10	**4**	1	1440	1440	1440	0.00	0.00	0.00	0.00	0.00	0.00
		2	1116	1116	1116	0.00	0.00				
		3	1008	1008	1008	0.00	0.00				
		4	1332	1332	1332	0.00	0.00				
		5	1404	1404	1404	0.00	0.00				
10	**6**	1	1120	1120	1120	0.00	0.00	0.00	0.00	0.00	0.00
		2	868	868	868	0.00	0.00				
		3	784	784	784	0.00	0.00				
		4	1036	1036	1036	0.00	0.00				
		5	1092	1092	1092	0.00	0.00				
10	**8**	1	960	960	960	0.00	0.00	0.00	0.00	0.00	0.00
		2	874	874	874	0.00	0.00				
		3	742	742	742	0.00	0.00				
		4	898	898	898	0.00	0.00				
		5	996	996	996	0.00	0.00				

Table C.3 : Computational results for *Last Trip Partial Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	*LastPartialHR1* (LPHR1)	*LastPartialHR2* (LPHR2)	Lower Bound	Lower Bound Gap (%)		Average Lower Bound Gap (%)		Maximum Lower Bound Gap (%)	
						LPHR1	LPHR2	LPHR1	LPHR2	LPHR1	LPHR2
25	**2**	1	13520	13520	13520	0.00	0.00	0.00	0.00	0.00	0.00
		2	10478	10478	10478	0.00	0.00				
		3	9464	9464	9464	0.00	0.00				
		4	12506	12506	12506	0.00	0.00				
		5	13182	13182	13182	0.00	0.00				
25	**4**	1	7280	7280	7280	0.00	0.00	0.00	0.00	0.00	0.00
		2	5642	5642	5642	0.00	0.00				
		3	5096	5096	5096	0.00	0.00				
		4	6734	6734	6734	0.00	0.00				
		5	7098	7098	7098	0.00	0.00				
25	**6**	1	5200	5200	5200	0.00	0.00	0.00	0.00	0.00	0.00
		2	4030	4030	4030	0.00	0.00				
		3	3640	3640	3640	0.00	0.00				
		4	4810	4810	4810	0.00	0.00				
		5	5070	5070	5070	0.00	0.00				
25	**8**	1	4160	4160	4160	0.00	0.00	0.33	0.33	1.67	1.67
		2	3278	3278	3224	1.67	1.67				
		3	2912	2912	2912	0.00	0.00				
		4	3848	3848	3848	0.00	0.00				
		5	4056	4056	4056	0.00	0.00				

Table C.3 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	*LastPartialHR1* (LPHR1)	*LastPartialHR2* (LPHR2)	Lower Bound	Lower Bound Gap (%)		Average Lower Bound Gap (%)		Maximum Lower Bound Gap (%)	
						LPHR1	LPHR2	LPHR1	LPHR2	LPHR1	LPHR2
40	**2**	1	33600	33600	33600	0.00	0.00	0.00	0.00	0.00	0.00
		2	26040	26040	26040	0.00	0.00				
		3	23520	23520	23520	0.00	0.00				
		4	31080	31080	31080	0.00	0.00				
		5	32760	32760	32760	0.00	0.00				
40	**4**	1	17600	17600	17600	0.00	0.00	0.00	0.00	0.00	0.00
		2	13640	13640	13640	0.00	0.00				
		3	12320	12320	12320	0.00	0.00				
		4	16280	16280	16280	0.00	0.00				
		5	17160	17160	17160	0.00	0.00				
40	**6**	1	12320	12320	12320	0.00	0.00	0.00	0.00	0.00	0.00
		2	9548	9548	9548	0.00	0.00				
		3	8624	8624	8624	0.00	0.00				
		4	11396	11396	11396	0.00	0.00				
		5	12012	12012	12012	0.00	0.00				
40	**8**	1	9600	9600	9600	0.00	0.00	0.02	0.19	0.12	0.95
		2	7440	7440	7440	0.00	0.00				
		3	6728	6784	6720	0.12	0.95				
		4	8880	8880	8880	0.00	0.00				
		5	9360	9360	9360	0.00	0.00				

Table C.3 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	*LastPartialHR1* (LPHR1)	*LastPartialHR2* (LPHR2)	Lower Bound	Lower Bound Gap (%)		Average Lower Bound Gap (%)		Maximum Lower Bound Gap (%)	
						LPHR1	LPHR2	LPHR1	LPHR2	LPHR1	LPHR2
55	**2**	1	62720	62720	62720	0.00	0.00	0.00	0.00	0.00	0.00
		2	48608	48608	48608	0.00	0.00				
		3	43904	43904	43904	0.00	0.00				
		4	58016	58016	58016	0.00	0.00				
		5	61152	61152	61152	0.00	0.00				
55	**4**	1	32480	32480	32480	0.00	0.00	0.00	0.00	0.00	0.00
		2	25172	25172	25172	0.00	0.00				
		3	22736	22736	22736	0.00	0.00				
		4	30044	30044	30044	0.00	0.00				
		5	31668	31668	31668	0.00	0.00				
55	**6**	1	22400	22400	22400	0.00	0.00	0.00	0.00	0.00	0.00
		2	17360	17360	17360	0.00	0.00				
		3	15680	15680	15680	0.00	0.00				
		4	20720	20720	20720	0.00	0.00				
		5	21840	21840	21840	0.00	0.00				
55	**8**	1	17360	17360	17360	0.00	0.00	0.00	0.00	0.00	0.00
		2	13454	13454	13454	0.00	0.00				
		3	12152	12152	12152	0.00	0.00				
		4	16058	16058	16058	0.00	0.00				
		5	16926	16926	16926	0.00	0.00				

Table C.3 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
10	**2**	1	1560	1560	1560	1560		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1440	1440	1440	1440		0.00	0.00	0.00	0.00								
		3	1380	1380	1380	1380		0.00	0.00	0.00	0.00								
		4	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
		5	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
10	**4**	1	936	936	936	936		0.00	0.00	0.00	0.00	0.13	0.27	0.67	1.34	0.00	0.13	0.00	0.67
		2	900	906	894	900	0.68	0.67	1.34	0.00	0.67								
		3	838	838	838	838		0.00	0.00	0.00	0.00								
		4	956	956	956	956		0.00	0.00	0.00	0.00								
		5	966	966	966	966		0.00	0.00	0.00	0.00								
10	**6**	1	898	898	898	898		0.00	0.00	0.00	0.00	0.37	0.37	1.00	1.00	0.00	0.00	0.00	0.00
		2	920	920	912	920	0.25	0.88	0.88	0.00	0.00								
		3	812	812	804	812	0.25	1.00	1.00	0.00	0.00								
		4	938	938	938	938		0.00	0.00	0.00	0.00								
		5	988	988	988	988		0.00	0.00	0.00	0.00								
10	**8**	1	1044	1044	1044	1044		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1096	1096	1096	1096		0.00	0.00	0.00	0.00								
		3	992	992	992	992		0.00	0.00	0.00	0.00								
		4	1064	1064	1064	1064		0.00	0.00	0.00	0.00								
		5	1134	1134	1134	1134		0.00	0.00	0.00	0.00								

Table C.4 : Computational results for *Last Trip Partial Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
25	**2**	1	8788	8788	8788			0.00	0.00			0.00	0.00	0.00	0.00				
		2	8112	8112	8112			0.00	0.00										
		3	7774	7774	7774			0.00	0.00										
		4	8788	8788	8788			0.00	0.00										
		5	8788	8788	8788			0.00	0.00										
25	**4**	1	4732	4732	4732			0.00	0.00			0.00	0.00	0.00	0.00				
		2	4443	4443	4443			0.00	0.00										
		3	4186	4186	4186			0.00	0.00										
		4	4732	4732	4732			0.00	0.00										
		5	4732	4732	4732			0.00	0.00										
25	**6**	1	3606	3606	3580			0.73	0.73			0.74	0.66	1.45	1.06				
		2	3647	3633	3595			1.45	1.06										
		3	3266	3266	3240			0.80	0.80										
		4	3706	3706	3680			0.71	0.71										
		5	3705	3705	3705			0.00	0.00										
25	**8**	1	3281	3263	3229			1.61	1.05			2.03	1.26	4.14	3.12				
		2	3502	3502	3396			3.12	3.12										
		3	3116	3026	2992			4.14	1.14										
		4	3413	3413	3379			1.01	1.01										
		5	3488	3479	3479			0.26	0.00										

Table C.4 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
40	**2**	1	21840	21840	21840			0.00	0.00			0.00	0.00	0.00	0.00				
		2	20160	20160	20160			0.00	0.00										
		3	19320	19320	19320			0.00	0.00										
		4	21840	21840	21840			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
40	**4**	1	11440	11440	11440			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10560	10560	10560			0.00	0.00										
		3	10120	10120	10120			0.00	0.00										
		4	11440	11440	11440			0.00	0.00										
		5	11440	11440	11440			0.00	0.00										
40	**6**	1	8280	8280	8248			0.39	0.39			0.55	0.55	1.02	1.02				
		2	7908	7908	7832			0.97	0.97										
		3	7520	7520	7444			1.02	1.02										
		4	8520	8520	8488			0.38	0.38										
		5	8408	8408	8408			0.00	0.00										
40	**8**	1	7056	7056	6960			1.38	1.38			1.51	1.12	2.63	1.50				
		2	6952	6896	6800			2.24	1.41										
		3	6568	6496	6400			2.63	1.50										
		4	7376	7376	7280			1.32	1.32										
		5	7200	7200	7200			0.00	0.00										

Table C.4 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
55	**2**	1	40768	40768	40768			0.00	0.00			0.00	0.00	0.00	0.00				
		2	37632	37632	37632			0.00	0.00										
		3	36064	36064	36064			0.00	0.00										
		4	40768	40768	40768			0.00	0.00										
		5	40768	40768	40768			0.00	0.00										
55	**4**	1	21112	21112	21112			0.00	0.00			0.00	0.00	0.00	0.00				
		2	19488	19488	19488			0.00	0.00										
		3	18676	18676	18676			0.00	0.00										
		4	21112	21112	21112			0.00	0.00										
		5	21112	21112	21112			0.00	0.00										
55	**6**	1	14954	14954	14890			0.43	0.43			0.52	0.46	1.08	0.80				
		2	14196	14157	14045			1.08	0.80										
		3	13408	13408	13320			0.66	0.66										
		4	15174	15174	15110			0.42	0.42										
		5	14945	14945	14945			0.00	0.00										
55	**8**	1	12475	12444	12274			1.64	1.39			1.22	1.18	1.68	1.68				
		2	11977	11977	11791			1.58	1.58										
		3	11268	11268	11082			1.68	1.68										
		4	12758	12760	12604			1.22	1.24										
		5	12329	12329	12329			0.00	0.00										

Table C.4 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	LB Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)	
						LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
10	**2**	1	2400	2400	2400	0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860	0.00	0.00				
		3	1680	1680	1680	0.00	0.00				
		4	2220	2220	2220	0.00	0.00				
		5	2340	2340	2340	0.00	0.00				
10	**4**	1	1440	1440	1440	0.00	0.00	0.00	0.00	0.00	0.00
		2	1116	1116	1116	0.00	0.00				
		3	1008	1008	1008	0.00	0.00				
		4	1332	1332	1332	0.00	0.00				
		5	1404	1404	1404	0.00	0.00				
10	**6**	1	1150	1150	1150	0.00	0.00	0.00	0.00	0.00	0.00
		2	1038	1038	1038	0.00	0.00				
		3	902	902	894	0.00	0.00				
		4	1136	1136	1136	0.00	0.00				
		5	1222	1222	1222	0.00	0.00				
10	**8**	1	1240	1240	1240	0.00	0.00	0.00	0.00	0.00	0.00
		2	1194	1194	1194	0.00	0.00				
		3	1062	1062	1062	0.00	0.00				
		4	1218	1218	1218	0.00	0.00				
		5	1316	1316	1316	0.00	0.00				

Table C.5 : Computational results for *Last Trip Partial Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	LB Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)	
						LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
25	**2**	1	13520	13520	13520	0.00	0.00	0.00	0.00	0.00	0.00
		2	10478	10478	10478	0.00	0.00				
		3	9464	9464	9464	0.00	0.00				
		4	12506	12506	12506	0.00	0.00				
		5	13182	13182	13182	0.00	0.00				
25	**4**	1	7280	7280	7280	0.00	0.00	0.00	0.00	0.00	0.00
		2	5642	5642	5642	0.00	0.00				
		3	5096	5096	5096	0.00	0.00				
		4	6734	6734	6734	0.00	0.00				
		5	7098	7098	7098	0.00	0.00				
25	**6**	1	5200	5200	5200	0.00	0.00	0.06	0.06	0.30	0.30
		2	4408	4408	4330	0.30	0.30				
		3	3821	3821	3765	0.00	0.00				
		4	4835	4835	4835	0.00	0.00				
		5	5070	5070	5070	0.00	0.00				
25	**8**	1	4371	4371	4335	0.85	0.85	0.67	0.67	0.90	0.90
		2	4051	4051	3949	0.81	0.81				
		3	3523	3523	3387	0.90	0.90				
		4	4337	4384	4248	0.82	0.82				
		5	4506	4506	4506	0.00	0.00				

Table C.5 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	LB Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)	
						LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
40	**2**	1	33600	33600	33600	0.00	0.00	0.00	0.00	0.00	0.00
		2	26040	26040	26040	0.00	0.00				
		3	23520	23520	23520	0.00	0.00				
		4	31080	31080	31080	0.00	0.00				
		5	32760	32760	32760	0.00	0.00				
40	**4**	1	17600	17600	17600	0.00	0.00	0.00	0.00	0.00	0.00
		2	13640	13640	13640	0.00	0.00				
		3	12320	12320	12320	0.00	0.00				
		4	16280	16280	16280	0.00	0.00				
		5	17160	17160	17160	0.00	0.00				
40	**6**	1	12320	12320	12320	0.00	0.00	0.00	0.00	0.00	0.00
		2	9768	9768	9708	0.00	0.00				
		3	8912	8912	8784	0.00	0.00				
		4	11436	11436	11436	0.00	0.00				
		5	12012	12012	12012	0.00	0.00				
40	**8**	1	9792	9792	9760	0.36	0.36	0.44	0.36	0.76	0.57
		2	8488	8440	8200	0.73	0.55				
		3	7816	7784	7400	0.76	0.57				
		4	9608	9608	9480	0.34	0.34				
		5	9800	9800	9800	0.00	0.00				

Table C.5 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	LB Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)	
						LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
55	**2**	1	62720	62720	62720	0.00	0.00	0.00	0.00	0.00	0.00
		2	48608	48608	48608	0.00	0.00				
		3	43904	43904	43904	0.00	0.00				
		4	58016	58016	58016	0.00	0.00				
		5	61152	61152	61152	0.00	0.00				
55	**4**	1	32480	32480	32480	0.00	0.00	0.00	0.00	0.00	0.00
		2	25172	25172	25172	0.00	0.00				
		3	22736	22736	22736	0.00	0.00				
		4	30044	30044	30044	0.00	0.00				
		5	31668	31668	31668	0.00	0.00				
55	**6**	1	22400	22400	22400	0.00	0.00	0.00	0.00	0.00	0.00
		2	17658	17658	17580	0.00	0.00				
		3	15949	15949	15845	0.00	0.00				
		4	20720	20720	20720	0.00	0.00				
		5	21840	21840	21840	0.00	0.00				
55	**8**	1	17640	17640	17580	0.29	0.29	0.31	0.27	0.59	0.49
		2	15001	14864	14444	0.59	0.49				
		3	13657	13409	12977	0.40	0.31				
		4	16985	16957	16773	0.28	0.28				
		5	17256	17256	17256	0.00	0.00				

Table C.5 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
10	**2**	1	2400	2400	2400	2400		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860	1860		0.00	0.00	0.00	0.00								
		3	1680	1680	1680	1680		0.00	0.00	0.00	0.00								
		4	2220	2220	2220	2220		0.00	0.00	0.00	0.00								
		5	2340	2340	2340	2340		0.00	0.00	0.00	0.00								
10	**4**	1	1440	1440	1440	1440		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1116	1116	1116	1116		0.00	0.00	0.00	0.00								
		3	1008	1008	1008	1008		0.00	0.00	0.00	0.00								
		4	1332	1332	1332	1332		0.00	0.00	0.00	0.00								
		5	1404	1404	1404	1404		0.00	0.00	0.00	0.00								
10	**6**	1	1150	1150	1150	1150		0.00	0.00	0.00	0.00	0.18	0.18	0.89	0.89	0.00	0.00	0.00	0.00
		2	1038	1038	1038	1038		0.00	0.00	0.00	0.00								
		3	902	902	894	902	0.10	0.89	0.89	0.00	0.00								
		4	1136	1136	1136	1136		0.00	0.00	0.00	0.00								
		5	1222	1222	1222	1222		0.00	0.00	0.00	0.00								
10	**8**	1	1240	1240	1240	1240		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1194	1194	1194	1194		0.00	0.00	0.00	0.00								
		3	1062	1062	1062	1062		0.00	0.00	0.00	0.00								
		4	1218	1218	1218	1218		0.00	0.00	0.00	0.00								
		5	1316	1316	1316	1316		0.00	0.00	0.00	0.00								

Table C.6 : Computational results for *Last Trip Partial Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
25	**2**	1	13520	13520	13520			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10478	10478	10478			0.00	0.00										
		3	9464	9464	9464			0.00	0.00										
		4	12506	12506	12506			0.00	0.00										
		5	13182	13182	13182			0.00	0.00										
25	**4**	1	7280	7280	7280			0.00	0.00			0.00	0.00	0.00	0.00				
		2	5642	5642	5642			0.00	0.00										
		3	5096	5096	5096			0.00	0.00										
		4	6734	6734	6734			0.00	0.00										
		5	7098	7098	7098			0.00	0.00										
25	**6**	1	5200	5200	5200			0.00	0.00			0.66	0.66	1.80	1.80				
		2	4408	4408	4330			1.80	1.80										
		3	3821	3821	3765			1.49	1.49										
		4	4835	4835	4835			0.00	0.00										
		5	5070	5070	5070			0.00	0.00										
25	**8**	1	4371	4371	4335			0.83	0.83			1.90	2.13	4.02	4.02				
		2	4051	4051	3949			2.58	2.58										
		3	3523	3523	3387			4.02	4.02										
		4	4337	4384	4248			2.10	3.20										
		5	4506	4506	4506			0.00	0.00										

Table C.6 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
40	**2**	1	33600	33600	33600			0.00	0.00			0.00	0.00	0.00	0.00				
		2	26040	26040	26040			0.00	0.00										
		3	23520	23520	23520			0.00	0.00										
		4	31080	31080	31080			0.00	0.00										
		5	32760	32760	32760			0.00	0.00										
40	**4**	1	17600	17600	17600			0.00	0.00			0.00	0.00	0.00	0.00				
		2	13640	13640	13640			0.00	0.00										
		3	12320	12320	12320			0.00	0.00										
		4	16280	16280	16280			0.00	0.00										
		5	17160	17160	17160			0.00	0.00										
40	**6**	1	12320	12320	12320			0.00	0.00			0.42	0.42	1.46	1.46				
		2	9768	9768	9708			0.62	0.62										
		3	8912	8912	8784			1.46	1.46										
		4	11436	11436	11436			0.00	0.00										
		5	12012	12012	12012			0.00	0.00										
40	**8**	1	9792	9792	9760			0.33	0.33			2.16	1.96	5.62	5.19				
		2	8488	8440	8200			3.51	2.93										
		3	7816	7784	7400			5.62	5.19										
		4	9608	9608	9480			1.35	1.35										
		5	9800	9800	9800			0.00	0.00										

Table C.6 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
55	**2**	1	62720	62720	62720			0.00	0.00			0.00	0.00	0.00	0.00				
		2	48608	48608	48608			0.00	0.00										
		3	43904	43904	43904			0.00	0.00										
		4	58016	58016	58016			0.00	0.00										
		5	61152	61152	61152			0.00	0.00										
55	**4**	1	32480	32480	32480			0.00	0.00			0.00	0.00	0.00	0.00				
		2	25172	25172	25172			0.00	0.00										
		3	22736	22736	22736			0.00	0.00										
		4	30044	30044	30044			0.00	0.00										
		5	31668	31668	31668			0.00	0.00										
55	**6**	1	22400	22400	22400			0.00	0.00			0.22	0.22	0.66	0.66				
		2	17658	17658	17580			0.44	0.44										
		3	15949	15949	15845			0.66	0.66										
		4	20720	20720	20720			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
55	**8**	1	17640	17640	17580			0.34	0.34			2.14	1.54	5.24	3.33				
		2	15001	14864	14444			3.86	2.91										
		3	13657	13409	12977			5.24	3.33										
		4	16985	16957	16773			1.26	1.10										
		5	17256	17256	17256			0.00	0.00										

Table C.6 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
10	**2**	1	1838	1826	1810	1820	4.24	1.55	0.88	0.99	0.33	0.96	0.73	1.55	0.96	0.40	0.18	0.99	0.33
		2	1730	1726	1710	1724	2.35	1.17	0.94	0.35	0.12								
		3	1680	1676	1660	1674	2.66	1.20	0.96	0.36	0.12								
		4	1846	1846	1830	1840	3.73	0.87	0.87	0.33	0.33								
		5	1810	1810	1810	1810	13.43	0.00	0.00	0.00	0.00								
10	**4**	1	1738	1738	1726	1736	4.05	0.70	0.70	0.12	0.12	0.56	0.56	0.73	0.73	0.09	0.09	0.12	0.12
		2	1706	1706	1694	1704	2.38	0.71	0.71	0.12	0.12								
		3	1650	1650	1638	1648	4.75	0.73	0.73	0.12	0.12								
		4	1768	1768	1756	1766	3.42	0.68	0.68	0.11	0.11								
		5	1766	1766	1766	1766	1.79	0.00	0.00	0.00	0.00								
10	**6**	1	2106	2106	2098	2106	0.57	0.38	0.38	0.00	0.00	0.31	0.31	0.40	0.40	0.00	0.00	0.00	0.00
		2	2120	2120	2112	2120	0.71	0.38	0.38	0.00	0.00								
		3	2012	2012	2004	2012	0.63	0.40	0.40	0.00	0.00								
		4	2146	2146	2138	2146	0.69	0.37	0.37	0.00	0.00								
		5	2188	2188	2188	2188	0.61	0.00	0.00	0.00	0.00								
10	**8**	1	2648	2648	2644	2648	0.14	0.15	0.15	0.00	0.00	0.12	0.12	0.15	0.15	0.00	0.00	0.00	0.00
		2	2700	2700	2696	2700	0.17	0.15	0.15	0.00	0.00								
		3	2596	2596	2592	2596	0.20	0.15	0.15	0.00	0.00								
		4	2668	2668	2664	2668	0.16	0.15	0.15	0.00	0.00								
		5	2734	2734	2734	2734	0.16	0.00	0.00	0.00	0.00								

Table C.7 : Computational results for *Last Trip Partial Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
25	**2**	1	9451	9444	9388			0.67	0.60			0.71	0.69	1.01	1.12				
		2	8876	8885	8787			1.01	1.12										
		3	8527	8543	8449			0.92	1.11										
		4	9528	9498	9438			0.95	0.64										
		5	9413	9413	9413			0.00	0.00										
25	**4**	1	6726	6718	6632			1.42	1.30			2.31	2.00	4.56	3.75				
		2	6679	6639	6443			3.66	3.04										
		3	6416	6366	6136			4.56	3.75										
		4	6776	6776	6682			1.41	1.41										
		5	6716	6716	6682			0.51	0.51										
25	**6**	1	6952	6693	6657			4.43	0.54			4.16	0.72	5.76	1.31				
		2	7115	6905	6833			4.13	1.05										
		3	6807	6520	6436			5.76	1.31										
		4	7031	6807	6771			3.84	0.53										
		5	7032	6864	6852			2.63	0.18										
25	**8**	1	7281	7281	7229			0.72	0.72			0.97	0.97	1.77	1.77				
		2	7502	7502	7396			1.43	1.43										
		3	7116	7116	6992			1.77	1.77										
		4	7431	7431	7379			0.70	0.70										
		5	7497	7497	7479			0.24	0.24										

Table C.7 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
40	**2**	1	22922	22916	22800			0.54	0.51			0.75	0.65	1.31	1.14				
		2	21404	21404	21200			0.96	0.96										
		3	20588	20632	20400			0.92	1.14										
		4	23180	23028	22880			1.31	0.65										
		5	22800	22800	22800			0.00	0.00										
40	**4**	1	14664	14616	14400			1.83	1.50			2.48	2.38	4.55	4.55				
		2	14192	14192	13680			3.74	3.74										
		3	13800	13800	13200			4.55	4.55										
		4	14800	14792	14560			1.65	1.59										
		5	14568	14552	14480			0.61	0.50										
40	**6**	1	13686	13368	13114			4.36	1.94			3.90	1.52	4.44	2.81				
		2	13696	13386	13138			4.25	1.89										
		3	13362	13154	12794			4.44	2.81										
		4	13904	13552	13464			3.27	0.65										
		5	13922	13538	13494			3.17	0.33										
40	**8**	1	15072	14616	14384			4.78	1.61			4.55	1.67	5.74	2.37				
		2	15328	14840	14496			5.74	2.37										
		3	14960	14528	14208			5.29	2.25										
		4	15312	14912	14736			3.91	1.19										
		5	15280	14968	14832			3.02	0.92										

Table C.7 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
55	**2**	1	42268	42320	42088			0.43	0.55			0.64	0.64	1.08	1.08				
		2	39366	39438	39062			0.78	0.96										
		3	37900	37953	37549			0.93	1.08										
		4	42652	42454	42198			1.08	0.61										
		5	42088	42088	42088			0.00	0.00										
55	**4**	1	25614	25592	25182			1.72	1.63			2.91	2.48	6.28	4.85				
		2	24726	24678	23778			3.99	3.79										
		3	24349	24023	22911			6.28	4.85										
		4	25928	25828	25402			2.07	1.68										
		5	25369	25353	25237			0.52	0.46										
55	**6**	1	23065	22384	21774			5.93	2.80			5.41	2.39	6.76	3.22				
		2	22971	22360	21878			5.00	2.20										
		3	22506	21758	21080			6.76	3.22										
		4	23042	22472	22141			4.07	1.49										
		5	23120	22445	21959			5.29	2.21										
55	**8**	1	24614	23749	23287			5.70	1.98			5.23	1.93	5.71	2.87				
		2	24590	23884	23425			4.97	1.96										
		3	23893	23251	22603			5.71	2.87										
		4	24844	23980	23651			5.04	1.39										
		5	24560	23798	23456			4.71	1.46										

Table C.7 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
10	**2**	1	2310	2310	2310	2310		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	2210	2210	2210	2210		0.00	0.00	0.00	0.00								
		3	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
		4	2330	2330	2330	2330		0.00	0.00	0.00	0.00								
		5	2310	2310	2310	2310		0.00	0.00	0.00	0.00								
10	**4**	1	1998	1998	1986	1996	2.48	0.60	0.60	0.10	0.10	0.49	0.49	0.63	0.63	0.08	0.08	0.10	0.10
		2	1966	1966	1954	1964	6.03	0.61	0.61	0.10	0.10								
		3	1910	1910	1898	1908	6.45	0.63	0.63	0.10	0.10								
		4	2028	2028	2016	2026	6.87	0.60	0.60	0.10	0.10								
		5	2026	2026	2026	2026		0.00	0.00	0.00	0.00								
10	**6**	1	2286	2286	2278	2286	0.34	0.35	0.35	0.00	0.00	0.28	0.28	0.37	0.37	0.00	0.00	0.00	0.00
		2	2300	2300	2292	2300	0.64	0.35	0.35	0.00	0.00								
		3	2192	2192	2184	2192	0.62	0.37	0.37	0.00	0.00								
		4	2326	2326	2318	2326	0.58	0.35	0.35	0.00	0.00								
		5	2368	2368	2368	2368		0.00	0.00	0.00	0.00								
10	**8**	1	2784	2784	2784	2784		0.00	0.00	0.00	0.00	0.06	0.06	0.15	0.15	0.00	0.00	0.00	0.00
		2	2840	2840	2836	2840	0.19	0.14	0.14	0.00	0.00								
		3	2736	2736	2732	2736	0.16	0.15	0.15	0.00	0.00								
		4	2804	2804	2804	2804		0.00	0.00	0.00	0.00								
		5	2874	2874	2874	2874		0.00	0.00	0.00	0.00								

Table C.8 : Computational results for *Last Trip Partial Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
25	**2**	1	12518	12518	12518			0.00	0.00			0.00	0.00	0.00	0.00				
		2	11917	11917	11917			0.00	0.00										
		3	11579	11579	11579			0.00	0.00										
		4	12568	12568	12568			0.00	0.00										
		5	12543	12543	12543			0.00	0.00										
25	**4**	1	8280	8270	8202			0.95	0.83			0.78	0.73	1.01	1.01				
		2	8091	8091	8013			0.97	0.97										
		3	7784	7784	7706			1.01	1.01										
		4	8330	8320	8252			0.95	0.82										
		5	8252	8252	8252			0.00	0.00										
25	**6**	1	7694	7694	7630			0.84	0.84			1.23	1.23	2.30	2.30				
		2	7787	7787	7645			1.86	1.86										
		3	7458	7458	7290			2.30	2.30										
		4	7794	7794	7730			0.83	0.83										
		5	7781	7781	7755			0.34	0.34										
25	**8**	1	8071	8071	8019			0.65	0.65			0.88	0.88	1.59	1.59				
		2	8292	8292	8186			1.29	1.29										
		3	7906	7906	7782			1.59	1.59										
		4	8221	8221	8169			0.64	0.64										
		5	8287	8287	8269			0.22	0.22										

Table C.8 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
40	**2**	1	30800	30800	30800			0.00	0.00			0.00	0.00	0.00	0.00				
		2	29200	29200	29200			0.00	0.00										
		3	28400	28400	28400			0.00	0.00										
		4	30880	30880	30880			0.00	0.00										
		5	30800	30800	30800			0.00	0.00										
40	**4**	1	18624	18584	18400			1.22	1.00			0.89	0.84	1.22	1.12				
		2	17880	17872	17680			1.13	1.09										
		3	17400	17392	17200			1.16	1.12										
		4	18736	18744	18560			0.95	0.99										
		5	18480	18480	18480			0.00	0.00										
40	**6**	1	15904	15892	15728			1.12	1.04			1.71	1.66	3.22	3.22				
		2	15716	15716	15312			2.64	2.64										
		3	15404	15404	14924			3.22	3.22										
		4	16144	16132	15968			1.10	1.03										
		5	15964	15944	15888			0.48	0.35										
40	**8**	1	16112	15496	15424			4.46	0.47			4.24	0.78	5.36	1.42				
		2	16368	15752	15536			5.36	1.39										
		3	16000	15464	15248			4.93	1.42										
		4	16352	15840	15776			3.65	0.41										
		5	16320	15904	15872			2.82	0.20										

Table C.8 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
55	**2**	1	57218	57218	57218			0.00	0.00			0.00	0.00	0.00	0.00				
		2	54192	54192	54192			0.00	0.00										
		3	52679	52679	52679			0.00	0.00										
		4	57328	57328	57328			0.00	0.00										
		5	57218	57218	57218			0.00	0.00										
55	**4**	1	33068	33086	32752			0.96	1.02			0.85	0.88	1.12	1.19				
		2	31688	31712	31348			1.08	1.16										
		3	30821	30845	30481			1.12	1.19										
		4	33323	33306	32972			1.06	1.01										
		5	32807	32807	32807			0.00	0.00										
55	**6**	1	26875	26850	26540			1.26	1.17			2.01	1.81	4.04	3.47				
		2	26481	26425	25695			3.06	2.84										
		3	25978	25836	24970			4.04	3.47										
		4	27070	27070	26760			1.16	1.16										
		5	26738	26707	26595			0.54	0.42										
55	**8**	1	26124	25181	24895			4.94	1.15			4.63	1.20	5.35	2.68				
		2	26100	25280	24970			4.53	1.24										
		3	25403	24759	24113			5.35	2.68										
		4	26354	25442	25287			4.22	0.61										
		5	26070	25128	25043			4.10	0.34										

Table C.8 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
10	**2**	1	2510	2510	2510	2510		0.00	0.00	0.00	0.00								
		2	2072	2072	2060	2072	5.77	0.58	0.58	0.00	0.00								
		3	1958	1942	1910	1942	4.06	2.51	1.68	0.82	0.00	0.62	0.45	2.51	1.68	0.16	0.00	0.82	0.00
		4	2380	2380	2380	2380		0.00	0.00	0.00	0.00								
		5	2460	2460	2460	2460		0.00	0.00	0.00	0.00								
10	**4**	1	2114	2114	2090	2110	5.59	1.15	1.15	0.19	0.19								
		2	1912	1912	1876	1906	3.14	1.92	1.92	0.31	0.31								
		3	1816	1816	1768	1808	1.53	2.71	2.71	0.44	0.44	1.63	1.63	2.71	2.71	0.27	0.27	0.44	0.44
		4	2090	2090	2042	2082	2.93	2.35	2.35	0.38	0.38								
		5	2104	2104	2104	2104		0.00	0.00	0.00	0.00								
10	**6**	1	2366	2366	2350	2366	0.73	0.68	0.68	0.00	0.00								
		2	2262	2262	2238	2262	0.63	1.07	1.07	0.00	0.00								
		3	2126	2126	2094	2126	0.75	1.53	1.53	0.00	0.00	0.93	0.93	1.53	1.53	0.00	0.00	0.00	0.00
		4	2368	2368	2336	2368	0.69	1.37	1.37	0.00	0.00								
		5	2422	2422	2422	2422		0.00	0.00	0.00	0.00								
10	**8**	1	2840	2840	2840	2840		0.00	0.00	0.00	0.00								
		2	2806	2806	2794	2806	0.19	0.43	0.43	0.00	0.00								
		3	2678	2678	2662	2678	0.17	0.60	0.60	0.00	0.00	0.21	0.21	0.60	0.60	0.00	0.00	0.00	0.00
		4	2818	2818	2818	2818		0.00	0.00	0.00	0.00								
		5	2916	2916	2916	2916		0.00	0.00	0.00	0.00								

Table C.9 : Computational results for *Last Trip Partial Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
25	**2**	1	13770	13770	13770			0.00	0.00			0.22	0.20	0.65	0.72				
		2	11026	11008	10978			0.44	0.27										
		3	10079	10086	10014			0.65	0.72										
		4	12881	12881	12881			0.00	0.00										
		5	13482	13482	13482			0.00	0.00										
25	**4**	1	8950	8950	8830			1.36	1.36			2.57	2.44	4.77	4.51				
		2	7776	7776	7542			3.10	3.10										
		3	7251	7233	6921			4.77	4.51										
		4	8714	8681	8409			3.63	3.23										
		5	8723	8723	8723			0.00	0.00										
25	**6**	1	8204	8204	8050			1.91	1.91			4.64	4.64	9.55	9.55				
		2	7834	7834	7330			6.88	6.88										
		3	7411	7411	6765			9.55	9.55										
		4	8091	8091	7835			3.27	3.27										
		5	8200	8200	8070			1.61	1.61										
25	**8**	1	8457	8457	8335			1.46	1.46			3.24	3.24	6.47	6.47				
		2	8321	8321	7949			4.68	4.68										
		3	7865	7865	7387			6.47	6.47										
		4	8456	8456	8248			2.52	2.52										
		5	8596	8596	8506			1.06	1.06										

Table C.9 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
40	**2**	1	34000	34000	34000			0.00	0.00			0.23	0.16	1.16	0.79				
		2	26800	26800	26800			0.00	0.00										
		3	24684	24592	24400			1.16	0.79										
		4	31680	31680	31680			0.00	0.00										
		5	33200	33200	33200			0.00	0.00										
40	**4**	1	20304	20288	20000			1.52	1.44			3.03	2.78	6.11	5.26				
		2	17080	17080	16480			3.64	3.64										
		3	16128	16000	15200			6.11	5.26										
		4	19696	19632	18960			3.88	3.54										
		5	19680	19680	19680			0.00	0.00										
40	**6**	1	17184	17120	16800			2.29	1.90			6.30	6.19	13.57	13.57				
		2	15948	15948	14508			9.93	9.93										
		3	15428	15428	13584			13.57	13.57										
		4	16892	16860	16236			4.04	3.84										
		5	16972	16972	16692			1.68	1.68										
40	**8**	1	16608	16464	16160			2.77	1.88			5.11	2.80	7.60	3.97				
		2	16328	15752	15224			7.25	3.47										
		3	15848	15312	14728			7.60	3.97										
		4	16648	16392	15880			4.84	3.22										
		5	16696	16440	16200			3.06	1.48										

Table C.9 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
55	**2**	1	63270	63270	63270			0.00	0.00			0.13	0.09	0.64	0.46				
		2	49653	49653	49653			0.00	0.00										
		3	45404	45322	45114			0.64	0.46										
		4	58841	58841	58841			0.00	0.00										
		5	61757	61757	61757			0.00	0.00										
55	**4**	1	36336	36308	35780			1.55	1.48			2.94	2.91	5.87	5.66				
		2	30211	30193	29077			3.90	3.84										
		3	28264	28208	26696			5.87	5.66										
		4	34873	34937	33729			3.39	3.58										
		5	35078	35078	35078			0.00	0.00										
55	**6**	1	29409	29158	28560			2.97	2.09			7.01	6.71	15.24	14.83				
		2	26679	26778	24180			10.33	10.74										
		3	25865	25773	22445			15.24	14.83										
		4	28505	28457	27265			4.55	4.37										
		5	28657	28540	28110			1.95	1.53										
55	**8**	1	26984	26984	26380			2.29	2.29			4.87	3.58	8.26	5.50				
		2	26127	25507	24482			6.72	4.19										
		3	25288	24642	23358			8.26	5.50										
		4	26793	26565	25573			4.77	3.88										
		5	26658	26596	26056			2.31	2.07										

Table C.9 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
10	**2**	1	1560	1560	1560	1560		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1440	1440	1440	1440		0.00	0.00	0.00	0.00								
		3	1380	1380	1380	1380		0.00	0.00	0.00	0.00								
		4	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
		5	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
10	**4**	1	966	966	936	958	0.95	3.21	3.21	0.84	0.84	6.48	6.48	17.07	17.07	1.04	1.04	2.27	2.27
		2	1070	1070	914	1048	2.12	17.07	17.07	2.10	2.10								
		3	900	900	828	880	0.64	8.70	8.70	2.27	2.27								
		4	1028	1028	1016	1028	2.83	1.18	1.18	0.00	0.00								
		5	1090	1090	1066	1090	1.42	2.25	2.25	0.00	0.00								
10	**6**	1	1030	1030	998	1030	0.15	3.21	3.21	0.00	0.00	5.62	5.62	19.18	19.18	0.00	0.00	0.00	0.00
		2	1178	1178	1122	1178	0.19	4.99	4.99	0.00	0.00								
		3	994	994	834	994	0.16	19.18	19.18	0.00	0.00								
		4	1136	1136	1128	1136	0.15	0.71	0.71	0.00	0.00								
		5	1318	1318	1318	1318		0.00	0.00	0.00	0.00								
10	**8**	1	1428	1428	1424	1428	0.15	0.28	0.28	0.00	0.00	0.22	0.22	0.30	0.30	0.00	0.00	0.00	0.00
		2	1650	1650	1646	1650	0.12	0.24	0.24	0.00	0.00								
		3	1336	1336	1332	1336	0.17	0.30	0.30	0.00	0.00								
		4	1478	1478	1474	1478	0.10	0.27	0.27	0.00	0.00								
		5	1704	1704	1704	1704		0.00	0.00	0.00	0.00								

Table C.10: Computational results for *Last Trip Partial Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
25	**2**	1	8788	8788	8788			0.00	0.00			0.00	0.00	0.00	0.00				
		2	8112	8112	8112			0.00	0.00										
		3	7774	7774	7774			0.00	0.00										
		4	8788	8788	8788			0.00	0.00										
		5	8788	8788	8788			0.00	0.00										
25	**4**	1	4760	4760	4732			0.59	0.59			1.00	0.79	2.60	2.40				
		2	4610	4601	4493			2.60	2.40										
		3	4214	4204	4186			0.67	0.43										
		4	4786	4758	4732			1.14	0.55										
		5	4732	4732	4732			0.00	0.00										
25	**6**	1	3660	3612	3380			8.28	6.86			9.13	7.64	15.22	12.77				
		2	4324	4336	3845	4296(*)		12.46	12.77										
		3	3445	3361	2990	3289(*)		15.22	12.41										
		4	3867	3757	3705			4.37	1.40										
		5	3954	3933	3755			5.30	4.74										
25	**8**	1	4310	4310	3079	4263	6898.34	39.98	39.98	1.10	1.10	29.44	29.44	41.89	41.89	0.91	0.91	1.58	1.58
		2	5052	5052	3896	5014(-)	50000.00	29.67	29.67	0.76	0.76								
		3	3997	3997	2817	3935	36647.46	41.89	41.89	1.58	1.58								
		4	4162	4162	3529	4130	12309.69	17.94	17.94	0.77	0.77								
		5	4537	4537	3854	4521	11960.39	17.72	17.72	0.35	0.35								

Table C.10 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

(-): The lower bound obtained by the linear relaxation is 5013.33. Best integer solution obtained is 5014.

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
40	**2**	1	21840	21840	21840			0.00	0.00			0.00	0.00	0.00	0.00				
		2	20160	20160	20160			0.00	0.00										
		3	19320	19320	19320			0.00	0.00										
		4	21840	21840	21840			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
40	**4**	1	11488	11488	11440			0.42	0.42			0.55	0.45	1.06	0.71				
		2	10672	10632	10560			1.06	0.68										
		3	10200	10192	10120			0.79	0.71										
		4	11496	11488	11440			0.49	0.42										
		5	11440	11440	11440			0.00	0.00										
40	**6**	1	8116	8116	8008			1.35	1.35			4.30	2.78	8.33	5.36				
		2	8230	8020	7632			7.84	5.08										
		3	7674	7464	7084			8.33	5.36										
		4	8548	8540	8408			1.67	1.57										
		5	8358	8212	8168			2.33	0.54										
40	**8**	1	7680	7680	6280			22.29	22.29			25.27	24.86	39.59	39.05				
		2	8640	8656	6800			27.06	27.29										
		3	8208	8176	5880			39.59	39.05										
		4	8328	8240	7400			12.54	11.35										
		5	8792	8752	7040			24.89	24.32										

Table C.10 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
55	**2**	1	40768	40768	40768			0.00	0.00			0.00	0.00	0.00	0.00				
		2	37632	37632	37632			0.00	0.00										
		3	36064	36064	36064			0.00	0.00										
		4	40768	40768	40768			0.00	0.00										
		5	40768	40768	40768			0.00	0.00										
55	**4**	1	21216	21202	21112			0.49	0.43			0.44	0.41	0.85	0.78				
		2	19654	19640	19488			0.85	0.78										
		3	18758	18758	18676			0.44	0.44										
		4	21202	21202	21112			0.43	0.43										
		5	21112	21112	21112			0.00	0.00										
55	**6**	1	14774	14736	14560			1.47	1.21			2.42	1.79	4.56	3.47				
		2	14398	14248	13770			4.56	3.47										
		3	13324	13204	12880			3.45	2.52										
		4	15104	15004	14780			2.19	1.52										
		5	14624	14598	14560			0.44	0.26										
55	**8**	1	13032	13107	11284			15.49	16.16			15.16	13.95	21.94	17.93				
		2	13684	13684	11736			16.60	16.60										
		3	12172	11772	9982			21.94	17.93										
		4	13530	13195	12384			9.25	6.55										
		5	12881	12881	11449			12.51	12.51										

Table C.10 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
10	**2**	1	2160	2160	2160	2160		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	2040	2040	2040	2040		0.00	0.00	0.00	0.00								
		3	1980	1980	1980	1980		0.00	0.00	0.00	0.00								
		4	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
		5	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
10	**4**	1	1296	1296	1296	1296		0.00	0.00	0.00	0.00	0.65	0.58	1.35	1.01	0.07	0.00	0.33	0.00
		2	1236	1236	1224	1236	0.45	0.98	0.98	0.00	0.00								
		3	1204	1200	1188	1200	0.16	1.35	1.01	0.33	0.00								
		4	1308	1308	1296	1308	0.11	0.93	0.93	0.00	0.00								
		5	1326	1326	1326	1326		0.00	0.00	0.00	0.00								
10	**6**	1	1186	1186	1178	1186	0.16	0.68	0.68	0.00	0.00	1.96	1.96	7.89	7.89	0.00	0.00	0.00	0.00
		2	1310	1310	1302	1310	0.17	0.61	0.61	0.00	0.00								
		3	1094	1094	1014	1094	0.16	7.89	7.89	0.00	0.00								
		4	1316	1316	1308	1316	0.14	0.61	0.61	0.00	0.00								
		5	1498	1498	1498	1498		0.00	0.00	0.00	0.00								
10	**8**	1	1564	1564	1564	1564		0.00	0.00	0.00	0.00	0.05	0.05	0.27	0.27	0.00	0.00	0.00	0.00
		2	1786	1786	1786	1786		0.00	0.00	0.00	0.00								
		3	1476	1476	1472	1476	0.08	0.27	0.27	0.00	0.00								
		4	1614	1614	1614	1614		0.00	0.00	0.00	0.00								
		5	1844	1844	1844	1844		0.00	0.00	0.00	0.00								

Table C.11: Computational results for *Last Trip Partial Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
25	**2**	1	12168	12168	12168			0.00	0.00			0.00	0.00	0.00	0.00				
		2	11492	11492	11492			0.00	0.00										
		3	11154	11154	11154			0.00	0.00										
		4	12168	12168	12168			0.00	0.00										
		5	12168	12168	12168			0.00	0.00										
25	**4**	1	6562	6562	6552			0.15	0.15			0.21	0.21	0.58	0.58				
		2	6224	6224	6188			0.58	0.58										
		3	6016	6016	6006			0.17	0.17										
		4	6562	6562	6552			0.15	0.15										
		5	6552	6552	6552			0.00	0.00										
25	**6**	1	4720	4706	4680			0.85	0.56			1.43	1.31	2.90	2.90				
		2	5037	5037	4895			2.90	2.90										
		3	4386	4386	4290			2.24	2.24										
		4	4795	4781	4755			0.84	0.55										
		5	4819	4819	4805			0.29	0.29										
25	**8**	1	4580	4580	3869			18.38	18.38			12.15	12.13	18.38	18.38				
		2	5322	5322	4686			13.57	13.57										
		3	4267	4263	3607			18.30	18.19										
		4	4488	4488	4319			3.91	3.91										
		5	4951	4951	4644			6.61	6.61										

Table C.11 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
40	**2**	1	30240	30240	30240			0.00	0.00			0.00	0.00	0.00	0.00				
		2	28560	28560	28560			0.00	0.00										
		3	27720	27720	27720			0.00	0.00										
		4	30240	30240	30240			0.00	0.00										
		5	30240	30240	30240			0.00	0.00										
40	**4**	1	15848	15848	15840			0.05	0.05			0.09	0.09	0.21	0.21				
		2	14992	14992	14960			0.21	0.21										
		3	14536	14536	14520			0.11	0.11										
		4	15848	15848	15840			0.05	0.05										
		5	15840	15840	15840			0.00	0.00										
40	**6**	1	11140	11140	11088			0.47	0.47			0.73	0.52	1.60	0.92				
		2	10640	10568	10472			1.60	0.92										
		3	10248	10228	10164			0.83	0.63										
		4	11152	11152	11088			0.58	0.58										
		5	11108	11088	11088			0.18	0.00										
40	**8**	1	9032	8720	8640			4.54	0.93			5.70	3.52	9.09	7.68				
		2	9504	9224	8800			8.00	4.82										
		3	8640	8528	7920			9.09	7.68										
		4	9512	9512	9400			1.19	1.19										
		5	9552	9312	9040			5.66	3.01										

Table C.11 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
55	**2**	1	56448	56448	56448			0.00	0.00			0.00	0.00	0.00	0.00				
		2	53312	53312	53312			0.00	0.00										
		3	51744	51744	51744			0.00	0.00										
		4	56448	56448	56448			0.00	0.00										
		5	56448	56448	56448			0.00	0.00										
55	**4**	1	29254	29254	29232			0.08	0.08			0.07	0.07	0.16	0.16				
		2	27652	27652	27608			0.16	0.16										
		3	26810	26810	26796			0.05	0.05										
		4	29246	29246	29232			0.05	0.05										
		5	29232	29232	29232			0.00	0.00										
55	**6**	1	20286	20274	20160			0.63	0.57			0.55	0.47	1.08	0.79				
		2	19246	19190	19040			1.08	0.79										
		3	18570	18556	18480			0.49	0.41										
		4	20274	20274	20160			0.57	0.57										
		5	20160	20160	20160			0.00	0.00										
55	**8**	1	15808	15778	15624			1.18	0.99			1.70	1.34	3.23	2.28				
		2	16027	15880	15526			3.23	2.28										
		3	14659	14596	14322			2.35	1.91										
		4	16374	16374	16174			1.24	1.24										
		5	15700	15670	15624			0.49	0.29										

Table C.11 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
10	**2**	1	2400	2400	2400	2400		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860	1860		0.00	0.00	0.00	0.00								
		3	1680	1680	1680	1680		0.00	0.00	0.00	0.00								
		4	2220	2220	2220	2220		0.00	0.00	0.00	0.00								
		5	2340	2340	2340	2340		0.00	0.00	0.00	0.00								
10	**4**	1	1440	1440	1440	1440		0.00	0.00	0.00	0.00	2.24	4.18	6.45	16.13	0.00	1.82	0.00	9.09
		2	1188	1296	1116	1188	0.45	6.45	16.13	0.00	9.09								
		3	1056	1056	1008	1056	0.06	4.76	4.76	0.00	0.00								
		4	1332	1332	1332	1332		0.00	0.00	0.00	0.00								
		5	1404	1404	1404	1404		0.00	0.00	0.00	0.00								
10	**6**	1	1266	1266	1250	1266	0.25	1.28	1.28	0.00	0.00	3.98	3.98	14.29	14.29	0.00	0.00	0.00	0.00
		2	1272	1272	1248	1272	0.26	1.92	1.92	0.00	0.00								
		3	1056	1056	924	1056	0.11	14.29	14.29	0.00	0.00								
		4	1358	1358	1326	1358	0.18	2.41	2.41	0.00	0.00								
		5	1552	1552	1552	1552		0.00	0.00	0.00	0.00								
10	**8**	1	1620	1620	1620	1620		0.00	0.00	0.00	0.00	0.37	0.37	1.14	1.14	0.00	0.00	0.00	0.00
		2	1756	1756	1744	1756	0.10	0.69	0.69	0.00	0.00								
		3	1418	1418	1402	1418	0.15	1.14	1.14	0.00	0.00								
		4	1628	1628	1628	1628		0.00	0.00	0.00	0.00								
		5	1886	1886	1886	1886		0.00	0.00	0.00	0.00								

Table C.12: Computational results for *Last Trip Partial Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
25	**2**	1	13520	13520	13520			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10478	10478	10478			0.00	0.00										
		3	9464	9464	9464			0.00	0.00										
		4	12506	12506	12506			0.00	0.00										
		5	13182	13182	13182			0.00	0.00										
25	**4**	1	7280	7280	7280			0.00	0.00			0.60	0.62	2.22	2.34				
		2	5767	5774	5642			2.22	2.34										
		3	5136	5136	5096			0.78	0.78										
		4	6734	6734	6734			0.00	0.00										
		5	7098	7098	7098			0.00	0.00										
25	**6**	1	5252	5252	5200			1.00	1.00			6.37	5.13	15.28	11.48				
		2	5280	5084	4580	5045(*)		15.28	11.00										
		3	4086	4058	3640	4015	2965.63	12.25	11.48	1.77	1.07								
		4	5020	4964	4860			3.29	2.14										
		5	5120	5120	5120			0.00	0.00										
25	**8**	1	4714	4714	4185	4695	12449.50	12.64	12.64	0.40	0.40	15.39	15.50	33.31	33.84				
		2	5365	5365	4449	5326	27821.13	20.59	20.59	0.73	0.73								
		3	4282	4299	3212	4223	8393.50	33.31	33.84	1.40	1.80								
		4	4606	4606	4398			4.73	4.73										
		5	5158	5158	4881			5.68	5.68										

Table C.12 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
40	**2**	1	33600	33600	33600			0.00	0.00			0.00	0.00	0.00	0.00				
		2	26040	26040	26040			0.00	0.00										
		3	23520	23520	23520			0.00	0.00										
		4	31080	31080	31080			0.00	0.00										
		5	32760	32760	32760			0.00	0.00										
40	**4**	1	17600	17600	17600			0.00	0.00			0.59	0.51	1.47	1.30				
		2	13840	13784	13640			1.47	1.06										
		3	12480	12480	12320			1.30	1.30										
		4	16312	16312	16280			0.20	0.20										
		5	17160	17160	17160			0.00	0.00										
40	**6**	1	12400	12384	12320			0.65	0.52			3.46	3.43	8.49	8.49				
		2	10112	10112	9548			5.91	5.91										
		3	9356	9356	8624			8.49	8.49										
		4	11652	11652	11396			2.25	2.25										
		5	12012	12012	12012			0.00	0.00										
40	**8**	1	9792	9744	9600			2.00	1.50			10.24	9.96	25.00	27.56				
		2	9544	9176	8200			16.39	11.90										
		3	8600	8776	6880			25.00	27.56										
		4	10048	10048	9600			4.67	4.67										
		5	9944	10040	9640			3.15	4.15										

Table C.12 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

n	K	Ins. #	LPH1	LPH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2	LPH1	LPH2
55	**2**	1	62720	62720	62720			0.00	0.00			0.00	0.00	0.00	0.00				
		2	48608	48608	48608			0.00	0.00										
		3	43904	43904	43904			0.00	0.00										
		4	58016	58016	58016			0.00	0.00										
		5	61152	61152	61152			0.00	0.00										
55	**4**	1	32480	32480	32480			0.00	0.00			0.33	0.31	0.79	0.77				
		2	25370	25352	25172			0.79	0.72										
		3	22912	22912	22736			0.77	0.77										
		4	30068	30068	30044			0.08	0.08										
		5	31668	31668	31668			0.00	0.00										
55	**6**	1	22580	22552	22400			0.80	0.68			2.35	2.21	5.00	4.25				
		2	18228	18092	17360			5.00	4.22										
		3	16198	16346	15680			3.30	4.25										
		4	21266	21120	20720			2.64	1.93										
		5	21840	21840	21840			0.00	0.00										
55	**8**	1	17698	17698	17360			1.95	1.95			6.07	5.25	12.84	10.22				
		2	15764	15589	14389			9.56	8.34										
		3	13712	13394	12152			12.84	10.22										
		4	17404	17353	16553			5.14	4.83										
		5	17076	17076	16926			0.89	0.89										

Table C.12 (continued): Computational results for *Last Trip Partial Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

Appendix D

Numerical Results for Fully Loaded Trips Problem

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
10	**2**	1	1560	1560	1560	1560		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1440	1440	1440	1440		0.00	0.00	0.00	0.00								
		3	1380	1380	1380	1380		0.00	0.00	0.00	0.00								
		4	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
		5	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
10	**4**	1	936	936	936	936		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	864	864	864	864		0.00	0.00	0.00	0.00								
		3	828	828	828	828		0.00	0.00	0.00	0.00								
		4	936	936	936	936		0.00	0.00	0.00	0.00								
		5	936	936	936	936		0.00	0.00	0.00	0.00								
10	**6**	1	728	728	728	728		0.00	0.00	0.00	0.00	0.55	0.55	2.75	2.75	0.00	0.00	0.00	0.00
		2	672	672	672	672		0.00	0.00	0.00	0.00								
		3	644	644	644	644		0.00	0.00	0.00	0.00								
		4	728	728	728	728		0.00	0.00	0.00	0.00								
		5	748	748	728	748	0.61	2.75	2.75	0.00	0.00								
10	**8**	1	724	724	624	724	0.25	16.03	16.03	0.00	0.00	24.43	24.43	34.72	34.72	0.00	0.00	0.00	0.00
		2	776	776	576	776	0.20	34.72	34.72	0.00	0.00								
		3	672	672	552	672	0.18	21.74	21.74	0.00	0.00								
		4	744	744	624	744	0.20	19.23	19.23	0.00	0.00								
		5	814	814	624	814	0.23	30.45	30.45	0.00	0.00								

Table D.1 : Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
25	**2**	1	8788	8788	8788			0.00	0.00			0.00	0.00	0.00	0.00				
		2	8112	8112	8112			0.00	0.00										
		3	7774	7774	7774			0.00	0.00										
		4	8788	8788	8788			0.00	0.00										
		5	8788	8788	8788			0.00	0.00										
25	**4**	1	4732	4732	4732			0.00	0.00			0.00	0.00	0.00	0.00				
		2	4368	4368	4368			0.00	0.00										
		3	4186	4186	4186			0.00	0.00										
		4	4732	4732	4732			0.00	0.00										
		5	4732	4732	4732			0.00	0.00										
25	**6**	1	3380	3380	3380			0.00	0.00			0.09	0.09	0.45	0.45				
		2	3134	3134	3120			0.45	0.45										
		3	2990	2990	2990			0.00	0.00										
		4	3380	3380	3380			0.00	0.00										
		5	3380	3380	3380			0.00	0.00										
25	**8**	1	2722	2722	2704			0.67	0.67			1.49	1.49	5.37	5.37				
		2	2630	2630	2496			5.37	5.37										
		3	2410	2410	2392			0.75	0.75										
		4	2722	2722	2704			0.67	0.67										
		5	2704	2704	2704			0.00	0.00										

Table D.1 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
40	**2**	1	21840	21840	21840			0.00	0.00			0.00	0.00	0.00	0.00				
		2	20160	20160	20160			0.00	0.00										
		3	19320	19320	19320			0.00	0.00										
		4	21840	21840	21840			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
40	**4**	1	11440	11440	11440			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10560	10560	10560			0.00	0.00										
		3	10120	10120	10120			0.00	0.00										
		4	11440	11440	11440			0.00	0.00										
		5	11440	11440	11440			0.00	0.00										
40	**6**	1	8008	8008	8008			0.00	0.00			0.00	0.00	0.00	0.00				
		2	7392	7392	7392			0.00	0.00										
		3	7084	7084	7084			0.00	0.00										
		4	8008	8008	8008			0.00	0.00										
		5	8008	8008	8008			0.00	0.00										
40	**8**	1	6256	6256	6240			0.26	0.26			0.33	0.33	0.58	0.58				
		2	5792	5792	5760			0.56	0.56										
		3	5552	5552	5520			0.58	0.58										
		4	6256	6256	6240			0.26	0.26										
		5	6240	6240	6240			0.00	0.00										

Table D.1 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
55	**2**	1	40768	40768	40768			0.00	0.00			0.00	0.00	0.00	0.00				
		2	37632	37632	37632			0.00	0.00										
		3	36064	36064	36064			0.00	0.00										
		4	40768	40768	40768			0.00	0.00										
		5	40768	40768	40768			0.00	0.00										
55	**4**	1	21112	21112	21112			0.00	0.00			0.00	0.00	0.00	0.00				
		2	19488	19488	19488			0.00	0.00										
		3	18676	18676	18676			0.00	0.00										
		4	21112	21112	21112			0.00	0.00										
		5	21112	21112	21112			0.00	0.00										
55	**6**	1	14560	14560	14560			0.00	0.00			0.00	0.00	0.00	0.00				
		2	13440	13440	13440			0.00	0.00										
		3	12880	12880	12880			0.00	0.00										
		4	14560	14560	14560			0.00	0.00										
		5	14560	14560	14560			0.00	0.00										
55	**8**	1	11298	11298	11284			0.12	0.12			0.14	0.14	0.29	0.29				
		2	10446	10446	10416			0.29	0.29										
		3	9996	9996	9982			0.14	0.14										
		4	11298	11298	11284			0.12	0.12										
		5	11284	11284	11284			0.00	0.00										

Table D.1 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
10	**2**	1	2160	2160	2160	2160		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	2040	2040	2040	2040		0.00	0.00	0.00	0.00								
		3	1980	1980	1980	1980		0.00	0.00	0.00	0.00								
		4	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
		5	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
10	**4**	1	1296	1296	1296	1296		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1224	1224	1224	1224		0.00	0.00	0.00	0.00								
		3	1188	1188	1188	1188		0.00	0.00	0.00	0.00								
		4	1296	1296	1296	1296		0.00	0.00	0.00	0.00								
		5	1296	1296	1296	1296		0.00	0.00	0.00	0.00								
10	**6**	1	1008	1008	1008	1008		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	952	952	952	952		0.00	0.00	0.00	0.00								
		3	924	924	924	924		0.00	0.00	0.00	0.00								
		4	1008	1008	1008	1008		0.00	0.00	0.00	0.00								
		5	1008	1008	1008	1008		0.00	0.00	0.00	0.00								
10	**8**	1	864	864	864	864		0.00	0.00	0.00	0.00	5.50	5.50	12.25	12.25	0.00	0.00	0.00	0.00
		2	916	916	816	916	0.17	12.25	12.25	0.00	0.00								
		3	812	812	792	812	0.15	2.53	2.53	0.00	0.00								
		4	884	884	864	884	0.04	2.31	2.31	0.00	0.00								
		5	954	954	864	954	0.18	10.42	10.42	0.00	0.00								

Table D.2: Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
25	**2**	1	12168	12168	12168			0.00	0.00			0.00	0.00	0.00	0.00				
		2	11492	11492	11492			0.00	0.00										
		3	11154	11154	11154			0.00	0.00										
		4	12168	12168	12168			0.00	0.00										
		5	12168	12168	12168			0.00	0.00										
25	**4**	1	6552	6552	6552			0.00	0.00			0.00	0.00	0.00	0.00				
		2	6188	6188	6188			0.00	0.00										
		3	6006	6006	6006			0.00	0.00										
		4	6552	6552	6552			0.00	0.00										
		5	6552	6552	6552			0.00	0.00										
25	**6**	1	4680	4680	4680			0.00	0.00			0.00	0.00	0.00	0.00				
		2	4420	4420	4420			0.00	0.00										
		3	4290	4290	4290			0.00	0.00										
		4	4680	4680	4680			0.00	0.00										
		5	4680	4680	4680			0.00	0.00										
25	**8**	1	3744	3744	3744			0.00	0.00			0.05	0.10	0.25	0.51				
		2	3545	3554	3536			0.25	0.51										
		3	3432	3432	3432			0.00	0.00										
		4	3744	3744	3744			0.00	0.00										
		5	3744	3744	3744			0.00	0.00										

Table D.2 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
40	**2**	1	30240	30240	30240			0.00	0.00			0.00	0.00	0.00	0.00				
		2	28560	28560	28560			0.00	0.00										
		3	27720	27720	27720			0.00	0.00										
		4	30240	30240	30240			0.00	0.00										
		5	30240	30240	30240			0.00	0.00										
40	**4**	1	15840	15840	15840			0.00	0.00			0.00	0.00	0.00	0.00				
		2	14960	14960	14960			0.00	0.00										
		3	14520	14520	14520			0.00	0.00										
		4	15840	15840	15840			0.00	0.00										
		5	15840	15840	15840			0.00	0.00										
40	**6**	1	11088	11088	11088			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10472	10472	10472			0.00	0.00										
		3	10164	10164	10164			0.00	0.00										
		4	11088	11088	11088			0.00	0.00										
		5	11088	11088	11088			0.00	0.00										
40	**8**	1	8640	8640	8640			0.00	0.00			0.00	0.00	0.00	0.00				
		2	8160	8160	8160			0.00	0.00										
		3	7920	7920	7920			0.00	0.00										
		4	8640	8640	8640			0.00	0.00										
		5	8640	8640	8640			0.00	0.00										

Table D.2 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
55	**2**	1	56448	56448	56448			0.00	0.00			0.00	0.00	0.00	0.00				
		2	53312	53312	53312			0.00	0.00										
		3	51744	51744	51744			0.00	0.00										
		4	56448	56448	56448			0.00	0.00										
		5	56448	56448	56448			0.00	0.00										
55	**4**	1	29232	29232	29232			0.00	0.00			0.00	0.00	0.00	0.00				
		2	27608	27608	27608			0.00	0.00										
		3	26796	26796	26796			0.00	0.00										
		4	29232	29232	29232			0.00	0.00										
		5	29232	29232	29232			0.00	0.00										
55	**6**	1	20160	20160	20160			0.00	0.00			0.00	0.00	0.00	0.00				
		2	19040	19040	19040			0.00	0.00										
		3	18480	18480	18480			0.00	0.00										
		4	20160	20160	20160			0.00	0.00										
		5	20160	20160	20160			0.00	0.00										
55	**8**	1	15624	15624	15624			0.00	0.00			0.00	0.00	0.00	0.00				
		2	14756	14756	14756			0.00	0.00										
		3	14322	14322	14322			0.00	0.00										
		4	15624	15624	15624			0.00	0.00										
		5	15624	15624	15624			0.00	0.00										

Table D.2 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
10	**2**	1	2400	2400	2400	2400		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860	1860		0.00	0.00	0.00	0.00								
		3	1680	1680	1680	1680		0.00	0.00	0.00	0.00								
		4	2220	2220	2220	2220		0.00	0.00	0.00	0.00								
		5	2340	2340	2340	2340		0.00	0.00	0.00	0.00								
10	**4**	1	1440	1440	1440	1440		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1116	1116	1116	1116		0.00	0.00	0.00	0.00								
		3	1008	1008	1008	1008		0.00	0.00	0.00	0.00								
		4	1332	1332	1332	1332		0.00	0.00	0.00	0.00								
		5	1404	1404	1404	1404		0.00	0.00	0.00	0.00								
10	**6**	1	1120	1120	1120	1120		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	868	868	868	868		0.00	0.00	0.00	0.00								
		3	784	784	784	784		0.00	0.00	0.00	0.00								
		4	1036	1036	1036	1036		0.00	0.00	0.00	0.00								
		5	1092	1092	1092	1092		0.00	0.00	0.00	0.00								
10	**8**	1	960	960	960	960		0.00	0.00	0.00	0.00	7.09	7.09	17.47	17.47	0.00	0.00	0.00	0.00
		2	874	874	744	874	0.17	17.47	17.47	0.00	0.00								
		3	742	742	672	742	0.21	10.42	10.42	0.00	0.00								
		4	898	898	888	898	0.10	1.13	1.13	0.00	0.00								
		5	996	996	936	996	0.13	6.41	6.41	0.00	0.00								

Table D.3: Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
25	**2**	1	13520	13520	13520			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10478	10478	10478			0.00	0.00										
		3	9464	9464	9464			0.00	0.00										
		4	12506	12506	12506			0.00	0.00										
		5	13182	13182	13182			0.00	0.00										
25	**4**	1	7280	7280	7280			0.00	0.00			0.00	0.00	0.00	0.00				
		2	5642	5642	5642			0.00	0.00										
		3	5096	5096	5096			0.00	0.00										
		4	6734	6734	6734			0.00	0.00										
		5	7098	7098	7098			0.00	0.00										
25	**6**	1	5200	5200	5200			0.00	0.00			0.00	0.00	0.00	0.00				
		2	4030	4030	4030			0.00	0.00										
		3	3640	3640	3640			0.00	0.00										
		4	4810	4810	4810			0.00	0.00										
		5	5070	5070	5070			0.00	0.00										
25	**8**	1	4160	4160	4160			0.00	0.00			0.33	0.33	1.67	1.67				
		2	3278	3278	3224			1.67	1.67										
		3	2912	2912	2912			0.00	0.00										
		4	3848	3848	3848			0.00	0.00										
		5	4056	4056	4056			0.00	0.00										

Table D.3 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
40	**2**	1	33600	33600	33600			0.00	0.00			0.00	0.00	0.00	0.00				
		2	26040	26040	26040			0.00	0.00										
		3	23520	23520	23520			0.00	0.00										
		4	31080	31080	31080			0.00	0.00										
		5	32760	32760	32760			0.00	0.00										
40	**4**	1	17600	17600	17600			0.00	0.00			0.00	0.00	0.00	0.00				
		2	13640	13640	13640			0.00	0.00										
		3	12320	12320	12320			0.00	0.00										
		4	16280	16280	16280			0.00	0.00										
		5	17160	17160	17160			0.00	0.00										
40	**6**	1	12320	12320	12320			0.00	0.00			0.00	0.00	0.00	0.00				
		2	9548	9548	9548			0.00	0.00										
		3	8624	8624	8624			0.00	0.00										
		4	11396	11396	11396			0.00	0.00										
		5	12012	12012	12012			0.00	0.00										
40	**8**	1	9600	9600	9600			0.00	0.00			0.02	0.19	0.12	0.95				
		2	7440	7440	7440			0.00	0.00										
		3	6728	6784	6720			0.12	0.95										
		4	8880	8880	8880			0.00	0.00										
		5	9360	9360	9360			0.00	0.00										

Table D.3 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
55	**2**	1	62720	62720	62720			0.00	0.00			0.00	0.00	0.00	0.00				
		2	48608	48608	48608			0.00	0.00										
		3	43904	43904	43904			0.00	0.00										
		4	58016	58016	58016			0.00	0.00										
		5	61152	61152	61152			0.00	0.00										
55	**4**	1	32480	32480	32480			0.00	0.00			0.00	0.00	0.00	0.00				
		2	25172	25172	25172			0.00	0.00										
		3	22736	22736	22736			0.00	0.00										
		4	30044	30044	30044			0.00	0.00										
		5	31668	31668	31668			0.00	0.00										
55	**6**	1	22400	22400	22400			0.00	0.00			0.00	0.00	0.00	0.00				
		2	17360	17360	17360			0.00	0.00										
		3	15680	15680	15680			0.00	0.00										
		4	20720	20720	20720			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
55	**8**	1	17360	17360	17360			0.00	0.00			0.00	0.00	0.00	0.00				
		2	13454	13454	13454			0.00	0.00										
		3	12152	12152	12152			0.00	0.00										
		4	16058	16058	16058			0.00	0.00										
		5	16926	16926	16926			0.00	0.00										

Table D.3 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
10	**2**	1	1560	1560	1560	1560		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1440	1440	1440	1440		0.00	0.00	0.00	0.00								
		3	1380	1380	1380	1380		0.00	0.00	0.00	0.00								
		4	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
		5	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
10	**4**	1	936	936	936	936		0.00	0.00	0.00	0.00	2.14	2.28	4.17	4.86	0.00	0.13	0.00	4.86
		2	900	906	864	900	16.86	4.17	4.86	0.00	0.67								
		3	838	838	828	838	14.64	1.21	1.21	0.00	0.00								
		4	956	956	936	956	7.60	2.14	2.14	0.00	0.00								
		5	966	966	936	966	4.27	3.21	3.21	0.00	0.00								
10	**6**	1	844	844	728	844	1.59	15.93	15.93	0.00	0.00	16.17	17.78	17.95	23.24	0.20	1.58	0.52	5.22
		2	828	846	702	826	1.08	17.95	20.51	0.24	2.42								
		3	770	806	654	766	0.61	17.74	23.24	0.52	5.22								
		4	864	864	748	862	1.15	15.51	15.51	0.23	0.23								
		5	862	862	758	862	1.06	13.72	13.72	0.00	0.00								
10	**8**	1	992	952	752	952	0.29	31.91	26.60	4.20	0.00	31.47	22.01	35.00	26.60	9.61	1.63	14.53	5.86
		2	1056	976	800	922	0.45	32.00	22.00	14.53	5.86								
		3	972	892	720	872	0.36	35.00	23.89	11.47	2.29								
		4	1008	952	768	952	0.40	31.25	23.96	5.88	0.00								
		5	1048	936	824	936	0.26	27.18	13.59	11.97	0.00								

Table D.4 : Computational results for *Fully Loaded Trips Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
25	**2**	1	8788	8788	8788			0.00	0.00			0.00	0.00	0.00	0.00				
		2	8112	8112	8112			0.00	0.00										
		3	7774	7774	7774			0.00	0.00										
		4	8788	8788	8788			0.00	0.00										
		5	8788	8788	8788			0.00	0.00										
25	**4**	1	4732	4732	4732			0.00	0.00			0.34	0.34	1.72	1.72				
		2	4443	4443	4368			1.72	1.72										
		3	4186	4186	4186			0.00	0.00										
		4	4732	4732	4732			0.00	0.00										
		5	4732	4732	4732			0.00	0.00										
25	**6**	1	3606	3606	3380			6.69	6.69			8.31	8.23	9.32	8.90				
		2	3647	3633	3336	3633(*)		9.32	8.90										
		3	3266	3266	3000			8.87	8.87										
		4	3706	3706	3408			8.74	8.74										
		5	3705	3705	3432			7.95	7.95										
25	**8**	1	3281	3263	3000	3263(*)		9.37	8.77			9.61	8.79	11.93	10.54				
		2	3502	3502	3168	3430(*)		10.54	10.54										
		3	3116	3026	2784	3026(*)		11.93	8.69										
		4	3413	3413	3144	3413(*)		8.56	8.56										
		5	3488	3479	3240	3479(*)		7.65	7.38										

Table D.4 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
40	**2**	1	21840	21840	21840			0.00	0.00			0.00	0.00	0.00	0.00				
		2	20160	20160	20160			0.00	0.00										
		3	19320	19320	19320			0.00	0.00										
		4	21840	21840	21840			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
40	**4**	1	11440	11440	11440			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10560	10560	10560			0.00	0.00										
		3	10120	10120	10120			0.00	0.00										
		4	11440	11440	11440			0.00	0.00										
		5	11440	11440	11440			0.00	0.00										
40	**6**	1	8280	8280	8008			3.40	3.40			5.58	5.58	6.98	6.98				
		2	7908	7908	7392			6.98	6.98										
		3	7520	7520	7084			6.15	6.15										
		4	8520	8520	8008			6.39	6.39										
		5	8408	8408	8008			5.00	5.00										
40	**8**	1	7056	7056	6960			1.38	1.38			1.51	1.12	2.63	1.50				
		2	6952	6896	6800			2.24	1.41										
		3	6568	6496	6400			2.63	1.50										
		4	7376	7376	7280			1.32	1.32										
		5	7200	7200	7200			0.00	0.00										

Table D.4 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
55	**2**	1	40768	40768	40768			0.00	0.00			0.00	0.00	0.00	0.00				
		2	37632	37632	37632			0.00	0.00										
		3	36064	36064	36064			0.00	0.00										
		4	40768	40768	40768			0.00	0.00										
		5	40768	40768	40768			0.00	0.00										
55	**4**	1	21112	21112	21112			0.00	0.00			0.00	0.00	0.00	0.00				
		2	19488	19488	19488			0.00	0.00										
		3	18676	18676	18676			0.00	0.00										
		4	21112	21112	21112			0.00	0.00										
		5	21112	21112	21112			0.00	0.00										
55	**6**	1	14954	14954	14560			2.71	2.71			3.65	3.59	4.74	4.45				
		2	14196	14157	13554			4.74	4.45										
		3	13408	13408	12880			4.10	4.10										
		4	15174	15174	14580			4.07	4.07										
		5	14945	14945	14560			2.64	2.64										
55	**8**	1	12416	12432	11944			3.95	4.09			3.82	3.85	4.50	4.35				
		2	11886	11886	11406			4.21	4.21										
		3	11236	11220	10752			4.50	4.35										
		4	12691	12707	12219			3.86	3.99										
		5	12311	12311	11999			2.60	2.60										

Table D.4 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
10	**2**	1	2160	2160	2160	2160		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	2040	2040	2040	2040		0.00	0.00	0.00	0.00								
		3	1980	1980	1980	1980		0.00	0.00	0.00	0.00								
		4	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
		5	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
10	**4**	1	1296	1296	1296	1296		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1224	1224	1224	1224		0.00	0.00	0.00	0.00								
		3	1188	1188	1188	1188		0.00	0.00	0.00	0.00								
		4	1296	1296	1296	1296		0.00	0.00	0.00	0.00								
		5	1296	1296	1296	1296		0.00	0.00	0.00	0.00								
10	**6**	1	1078	1078	1008	1078	0.92	6.94	6.94	0.00	0.00	10.67	10.67	14.71	14.71	0.00	0.00	0.00	0.00
		2	1092	1092	952	1092	1.67	14.71	14.71	0.00	0.00								
		3	984	984	924	984	1.17	6.49	6.49	0.00	0.00								
		4	1118	1118	1008	1118	0.88	10.91	10.91	0.00	0.00								
		5	1152	1152	1008	1152	1.01	14.29	14.29	0.00	0.00								
10	**8**	1	1184	1184	864	1184	0.17	37.04	37.04	0.00	0.00	39.85	39.85	41.50	41.50	0.00	0.00	0.00	0.00
		2	1236	1236	880	1236	0.92	40.45	40.45	0.00	0.00								
		3	1132	1132	800	1132	0.25	41.50	41.50	0.00	0.00								
		4	1204	1204	864	1204	0.26	39.35	39.35	0.00	0.00								
		5	1274	1274	904	1274	0.20	40.93	40.93	0.00	0.00								

Table D.5: Computational results for *Fully Loaded Trips Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
25	**2**	1	12168	12168	12168			0.00	0.00			0.00	0.00	0.00	0.00				
		2	11492	11492	11492			0.00	0.00										
		3	11154	11154	11154			0.00	0.00										
		4	12168	12168	12168			0.00	0.00										
		5	12168	12168	12168			0.00	0.00										
25	**4**	1	6552	6552	6552			0.00	0.00			0.00	0.00	0.00	0.00				
		2	6188	6188	6188			0.00	0.00										
		3	6006	6006	6006			0.00	0.00										
		4	6552	6552	6552			0.00	0.00										
		5	6552	6552	6552			0.00	0.00										
25	**6**	1	4680	4680	4680			0.00	0.00			1.62	1.62	5.41	5.41				
		2	4659	4659	4420			5.41	5.41										
		3	4290	4290	4290			0.00	0.00										
		4	4730	4730	4680			1.07	1.07										
		5	4755	4755	4680			1.60	1.60										
25	**8**	1	4053	4053	3744			8.25	8.25			8.45	8.45	8.90	8.90				
		2	4220	4220	3888	4220(*)		8.54	8.54										
		3	3816	3816	3504	3807(*)		8.90	8.90										
		4	4203	4203	3864	4194(*)		8.77	8.77										
		5	4269	4269	3960	4269(*)		7.80	7.80										

Table D.5 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
40	**2**	1	30240	30240	30240			0.00	0.00			0.00	0.00	0.00	0.00				
		2	28560	28560	28560			0.00	0.00										
		3	27720	27720	27720			0.00	0.00										
		4	30240	30240	30240			0.00	0.00										
		5	30240	30240	30240			0.00	0.00										
40	**4**	1	15840	15840	15840			0.00	0.00			0.00	0.00	0.00	0.00				
		2	14960	14960	14960			0.00	0.00										
		3	14520	14520	14520			0.00	0.00										
		4	15840	15840	15840			0.00	0.00										
		5	15840	15840	15840			0.00	0.00										
40	**6**	1	11088	11088	11088			0.00	0.00			0.22	0.22	0.72	0.72				
		2	10512	10512	10472			0.38	0.38										
		3	10164	10164	10164			0.00	0.00										
		4	11168	11168	11088			0.72	0.72										
		5	11088	11088	11088			0.00	0.00										
40	**8**	1	8992	8992	8960			0.36	0.36			0.44	0.36	0.76	0.57				
		2	8864	8848	8800			0.73	0.55										
		3	8464	8448	8400			0.76	0.57										
		4	9312	9312	9280			0.34	0.34										
		5	9200	9200	9200			0.00	0.00										

Table D.5 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
55	**2**	1	56448	56448	56448			0.00	0.00			0.00	0.00	0.00	0.00				
		2	53312	53312	53312			0.00	0.00										
		3	51744	51744	51744			0.00	0.00										
		4	56448	56448	56448			0.00	0.00										
		5	56448	56448	56448			0.00	0.00										
55	**4**	1	29232	29232	29232			0.00	0.00			0.00	0.00	0.00	0.00				
		2	27608	27608	27608			0.00	0.00										
		3	26796	26796	26796			0.00	0.00										
		4	29232	29232	29232			0.00	0.00										
		5	29232	29232	29232			0.00	0.00										
55	**6**	1	20160	20160	20160			0.00	0.00			0.06	0.06	0.29	0.29				
		2	19095	19095	19040			0.29	0.29										
		3	18480	18480	18480			0.00	0.00										
		4	20160	20160	20160			0.00	0.00										
		5	20160	20160	20160			0.00	0.00										
55	**8**	1	16110	16110	15734			2.39	2.39			2.60	2.56	3.14	3.03				
		2	15673	15657	15196			3.14	3.03										
		3	14932	14918	14542			2.68	2.59										
		4	16440	16440	16009			2.69	2.69										
		5	16119	16119	15789			2.09	2.09										

Table D.5 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
10	**2**	1	2400	2400	2400	2400		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860	1860		0.00	0.00	0.00	0.00								
		3	1680	1680	1680	1680		0.00	0.00	0.00	0.00								
		4	2220	2220	2220	2220		0.00	0.00	0.00	0.00								
		5	2340	2340	2340	2340		0.00	0.00	0.00	0.00								
10	**4**	1	1440	1440	1440	1440		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1116	1116	1116	1116		0.00	0.00	0.00	0.00								
		3	1008	1008	1008	1008		0.00	0.00	0.00	0.00								
		4	1332	1332	1332	1332		0.00	0.00	0.00	0.00								
		5	1404	1404	1404	1404		0.00	0.00	0.00	0.00								
10	**6**	1	1150	1150	1120	1150	1.08	2.68	2.68	0.00	0.00	11.54	11.54	18.43	18.43	0.00	0.00	0.00	0.00
		2	1028	1028	868	1028	1.25	18.43	18.43	0.00	0.00								
		3	902	902	784	902	0.70	15.05	15.05	0.00	0.00								
		4	1136	1136	1036	1136	0.92	9.65	9.65	0.00	0.00								
		5	1222	1222	1092	1222	1.29	11.90	11.90	0.00	0.00								
10	**8**	1	1240	1240	960	1240	0.24	29.17	29.17	0.00	0.00	36.53	36.53	40.60	40.60	0.00	0.00	0.00	0.00
		2	1164	1164	856	1164	0.28	35.98	35.98	0.00	0.00								
		3	1062	1062	760	1062	0.23	39.74	39.74	0.00	0.00								
		4	1218	1218	888	1218	0.22	37.16	37.16	0.00	0.00								
		5	1316	1316	936	1316	0.23	40.60	40.60	0.00	0.00								

Table D.6: Computational results for *Fully Loaded Trips Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
25	**2**	1	13520	13520	13520			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10478	10478	10478			0.00	0.00										
		3	9464	9464	9464			0.00	0.00										
		4	12506	12506	12506			0.00	0.00										
		5	13182	13182	13182			0.00	0.00										
25	**4**	1	7280	7280	7280			0.00	0.00			0.00	0.00	0.00	0.00				
		2	5642	5642	5642			0.00	0.00										
		3	5096	5096	5096			0.00	0.00										
		4	6734	6734	6734			0.00	0.00										
		5	7098	7098	7098			0.00	0.00										
25	**6**	1	5200	5200	5200			0.00	0.00			2.97	2.97	9.38	9.38				
		2	4408	4408	4030			9.38	9.38										
		3	3821	3821	3640			4.97	4.97										
		4	4835	4835	4810			0.52	0.52										
		5	5070	5070	5070			0.00	0.00										
25	**8**	1	4371	4371	4160			5.07	5.07			9.11	9.35	12.05	25.65				
		2	4051	4051	3672			10.32	10.32										
		3	3523	3523	3144			12.05	12.05										
		4	4337	4384	3936			10.19	11.38										
		5	4506	4506	4176			7.90	7.90										

Table D.6 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
40	**2**	1	33600	33600	33600			0.00	0.00			0.00	0.00	0.00	0.00				
		2	26040	26040	26040			0.00	0.00										
		3	23520	23520	23520			0.00	0.00										
		4	31080	31080	31080			0.00	0.00										
		5	32760	32760	32760			0.00	0.00										
40	**4**	1	17600	17600	17600			0.00	0.00			0.00	0.00	0.00	0.00				
		2	13640	13640	13640			0.00	0.00										
		3	12320	12320	12320			0.00	0.00										
		4	16280	16280	16280			0.00	0.00										
		5	17160	17160	17160			0.00	0.00										
40	**6**	1	12320	12320	12320			0.00	0.00			1.20	1.20	3.34	3.34				
		2	9768	9768	9548			2.30	2.30										
		3	8912	8912	8624			3.34	3.34										
		4	11436	11436	11396			0.35	0.35										
		5	12012	12012	12012			0.00	0.00										
40	**8**	1	9792	9792	9760			0.33	0.33			2.16	1.96	5.62	5.19				
		2	8488	8440	8200			3.51	2.93										
		3	7816	7784	7400			5.62	5.19										
		4	9608	9608	9480			1.35	1.35										
		5	9800	9800	9800			0.00	0.00										

Table D.6 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
55	**2**	1	62720	62720	62720			0.00	0.00			0.00	0.00	0.00	0.00				
		2	48608	48608	48608			0.00	0.00										
		3	43904	43904	43904			0.00	0.00										
		4	58016	58016	58016			0.00	0.00										
		5	61152	61152	61152			0.00	0.00										
55	**4**	1	32480	32480	32480			0.00	0.00			0.00	0.00	0.00	0.00				
		2	25172	25172	25172			0.00	0.00										
		3	22736	22736	22736			0.00	0.00										
		4	30044	30044	30044			0.00	0.00										
		5	31668	31668	31668			0.00	0.00										
55	**6**	1	22400	22400	22400			0.00	0.00			0.69	0.69	1.72	1.72				
		2	17658	17658	17360			1.72	1.72										
		3	15949	15949	15680			1.72	1.72										
		4	20720	20720	20720			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
55	**8**	1	17640	17640	17360			1.61	1.61			3.69	3.62	6.03	6.03				
		2	14815	14767	14059			5.38	5.04										
		3	13409	13409	12647			6.03	6.03										
		4	16960	16957	16388			3.49	3.47										
		5	17256	17256	16926			1.95	1.95										

Table D.6 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
10	**2**	1	1826	1826	1810	1820	784.62	0.88	0.88	0.33	0.33	0.83	0.73	1.20	0.96	0.27	0.18	0.36	0.33
		2	1730	1726	1710	1724	587.18	1.17	0.94	0.35	0.12								
		3	1680	1676	1660	1674	759.24	1.20	0.96	0.36	0.12								
		4	1846	1846	1830	1840	458.32	0.87	0.87	0.33	0.33								
		5	1810	1810	1810	1810		0.00	0.00	0.00	0.00								
10	**4**	1	1482	1482	1256	1452	21.96	17.99	17.99	2.07	2.07	19.23	19.23	21.77	21.77	3.15	3.15	4.57	4.57
		2	1510	1510	1240	1444	91.09	21.77	21.77	4.57	4.57								
		3	1428	1428	1200	1380	40.62	19.00	19.00	3.48	3.48								
		4	1502	1502	1280	1472	23.32	17.34	17.34	2.04	2.04								
		5	1546	1546	1288	1492	31.23	20.03	20.03	3.62	3.62								
10	**6**	1	1826	1826	1518	1792	1.36	20.29	20.29	1.90	1.90	21.53	21.53	24.48	24.48	2.06	2.06	2.49	2.49
		2	1868	1868	1502	1824	2.17	24.37	24.37	2.41	2.41								
		3	1810	1810	1454	1766	1.84	24.48	24.48	2.49	2.49								
		4	1844	1844	1548	1812	1.58	19.12	19.12	1.77	1.77								
		5	1860	1860	1558	1828	1.81	19.38	19.38	1.75	1.75								
10	**8**	1	2322	2322	2032	2220	0.8	14.27	14.27	4.59	4.59	14.39	14.39	15.40	15.40	4.77	4.77	5.48	5.48
		2	2390	2390	2080	2270	0.66	14.90	14.90	5.29	5.29								
		3	2308	2308	2000	2188	0.62	15.40	15.40	5.48	5.48								
		4	2342	2342	2048	2240	0.46	14.36	14.36	4.55	4.55								
		5	2378	2378	2104	2288	0.42	13.02	13.02	3.93	3.93								

Table D.7: Computational results for *Fully Loaded Trips Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
25	**2**	1	9451	9444	8788			7.54	7.46			7.78	7.80	8.16	8.33				
		2	8801	8809	8137			8.16	8.26										
		3	8460	8476	7824			8.13	8.33										
		4	9486	9476	8788			7.94	7.83										
		5	9412	9412	8788			7.10	7.10										
25	**4**	1	6348	6332	6192			2.52	2.26			3.63	3.34	5.46	5.06				
		2	6353	6329	6024			5.46	5.06										
		3	6048	6024	5736			5.44	5.02										
		4	6396	6404	6240			2.50	2.63										
		5	6380	6348	6240			2.24	1.73										
25	**6**	1	6604	6454	6258			5.53	3.13			5.73	3.34	7.24	4.60				
		2	6799	6715	6420			5.90	4.60										
		3	6486	6288	6048			7.24	3.97										
		4	6688	6508	6366			5.06	2.23										
		5	6754	6616	6438			4.91	2.76										
25	**8**	1	7020	6996	6840			2.63	2.28			3.05	3.06	5.04	4.58				
		2	7361	7329	7008			5.04	4.58										
		3	6840	6904	6624			3.26	4.23										
		4	7124	7124	6984			2.00	2.00										
		5	7244	7236	7080			2.32	2.20										

Table D.7 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
40	**2**	1	22922	22916	22800			0.54	0.51			0.57	0.65	0.92	1.14				
		2	21376	21404	21200			0.83	0.96										
		3	20588	20632	20400			0.92	1.14										
		4	23008	23028	22880			0.56	0.65										
		5	22800	22800	22800			0.00	0.00										
40	**4**	1	14664	14616	14400			1.83	1.50			2.48	2.38	4.55	4.55				
		2	14192	14192	13680			3.74	3.74										
		3	13800	13800	13200			4.55	4.55										
		4	14800	14792	14560			1.65	1.59										
		5	14568	14552	14480			0.61	0.50										
40	**6**	1	12714	12456	11248			13.03	10.74			13.52	11.34	15.37	13.90				
		2	12996	12708	11288			15.13	12.58										
		3	12684	12522	10994			15.37	13.90										
		4	12844	12580	11536			11.34	9.05										
		5	13040	12776	11570			12.71	10.42										
40	**8**	1	15072	14616	14384			4.78	1.61			4.55	1.67	5.74	2.37				
		2	15328	14840	14496			5.74	2.37										
		3	14960	14528	14208			5.29	2.25										
		4	15312	14912	14736			3.91	1.19										
		5	15280	14968	14832			3.02	0.92										

Table D.7 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
55	**2**	1	42268	42320	40768			3.68	3.81			3.87	4.00	4.31	4.45				
		2	39315	39371	37692			4.31	4.45										
		3	37744	37832	36234			4.17	4.41										
		4	42390	42436	40768			3.98	4.09										
		5	42088	42088	40768			3.24	3.24										
55	**4**	1	24931	24923	23807			4.72	4.69			5.61	5.62	8.18	8.13				
		2	23900	23956	22348			6.94	7.20										
		3	23298	23286	21536			8.18	8.13										
		4	25104	25096	23972			4.72	4.69										
		5	24635	24619	23807			3.48	3.41										
55	**6**	1	22498	21934	21114			6.55	3.88			6.18	3.65	7.54	4.67				
		2	22507	21985	21222			6.06	3.60										
		3	21990	21402	20448			7.54	4.67										
		4	22498	22072	21474			4.77	2.78										
		5	22570	22006	21294			5.99	3.34										
55	**8**	1	23884	23356	22201			7.58	5.20			7.19	5.09	7.70	6.25				
		2	23906	23490	22308			7.16	5.30										
		3	23192	22880	21534			7.70	6.25										
		4	24055	23447	22534			6.75	4.05										
		5	23859	23387	22346			6.77	4.66										

Table D.7 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
10	**2**	1	2310	2310	2310	2310		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	2210	2210	2210	2210		0.00	0.00	0.00	0.00								
		3	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
		4	2330	2330	2330	2330		0.00	0.00	0.00	0.00								
		5	2310	2310	2310	2310		0.00	0.00	0.00	0.00								
10	**4**	1	1750	1750	1446	1750	35.64	21.02	21.02	0.00	0.00	20.66	20.42	21.64	21.02	0.19	0.00	0.97	0.00
		2	1682	1682	1394	1682	35.67	20.14	20.14	0.00	0.00								
		3	1664	1648	1368	1648	59.81	21.64	20.47	0.97	0.00								
		4	1770	1770	1466	1770	33.15	20.74	20.74	0.00	0.00								
		5	1734	1734	1446	1734	19.95	19.75	19.75	0.00	0.00								
10	**6**	1	1926	1926	1698	1912	1.24	13.43	13.43	0.73	0.73	14.52	14.52	17.00	17.00	0.90	0.90	1.27	1.27
		2	1968	1968	1682	1944	1.78	17.00	17.00	1.23	1.23								
		3	1910	1910	1634	1886	1.55	16.89	16.89	1.27	1.27								
		4	1944	1944	1728	1932	1.34	12.50	12.50	0.62	0.62								
		5	1960	1960	1738	1948	1.13	12.77	12.77	0.62	0.62								
10	**8**	1	2422	2422	1014	2320	0.55	14.68	14.68	4.40	4.40	14.79	14.79	15.77	15.77	4.57	4.57	5.24	5.24
		2	2490	2490	986	2370	1.02	15.28	15.28	5.06	5.06								
		3	2408	2408	972	2288	1.15	15.77	15.77	5.24	5.24								
		4	2442	2442	1034	2340	0.91	14.76	14.76	4.36	4.36								
		5	2478	2478	1014	2388	0.95	13.46	13.46	3.77	3.77								

Table D.8: Computational results for *Fully Loaded Trips Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
25	**2**	1	12518	12518	12168			2.88	2.88			3.35	3.35	3.81	3.81				
		2	11917	11917	11492			3.70	3.70										
		3	11579	11579	11154			3.81	3.81										
		4	12568	12568	12168			3.29	3.29										
		5	12543	12543	12168			3.08	3.08										
25	**4**	1	7928	7936	6552			3.88	3.98			2.88	3.13	3.88	4.26				
		2	7676	7604	6188			2.84	1.88										
		3	7338	7482	6006			2.26	4.26										
		4	7928	7936	6552			3.23	3.33										
		5	7848	7848	6552			2.19	2.19										
25	**6**	1	7308	7284	4680			2.18	1.85			2.94	2.20	3.95	3.25				
		2	7448	7376	4420			3.79	2.79										
		3	7110	7062	4290			3.95	3.25										
		4	7404	7380	4680			2.15	1.82										
		5	7464	7368	4680			2.64	1.32										
25	**8**	1	7744	7672	3744			2.43	1.48			2.68	1.68	3.57	2.31				
		2	8004	7884	3548			3.57	2.02										
		3	7562	7514	3435			2.97	2.31										
		4	7864	7816	3744			2.08	1.45										
		5	7984	7888	3744			2.36	1.13										

Table D.8 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
40	**2**	1	30800	30800	30800			0.00	0.00			0.00	0.00	0.00	0.00				
		2	29200	29200	29200			0.00	0.00										
		3	28400	28400	28400			0.00	0.00										
		4	30880	30880	30880			0.00	0.00										
		5	30800	30800	30800			0.00	0.00										
40	**4**	1	18576	18584	18400			0.96	1.00			0.84	0.84	1.16	1.12				
		2	17880	17872	17680			1.13	1.09										
		3	17400	17392	17200			1.16	1.12										
		4	18736	18744	18560			0.95	0.99										
		5	18480	18480	18480			0.00	0.00										
40	**6**	1	14704	14692	13648			7.74	7.65			8.42	8.39	10.28	10.28				
		2	14452	14452	13192			9.55	9.55										
		3	14164	14164	12844			10.28	10.28										
		4	14864	14852	13808			7.65	7.56										
		5	14676	14676	13728			6.91	6.91										
40	**8**	1	16112	15496	15424			4.46	0.47			4.24	0.78	5.36	1.42				
		2	16368	15752	15536			5.36	1.39										
		3	16000	15464	15248			4.93	1.42										
		4	16352	15840	15776			3.65	0.41										
		5	16320	15904	15872			2.82	0.20										

Table D.8 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
55	**2**	1	57218	57218	56448			1.36	1.36			1.55	1.55	1.81	1.81				
		2	54192	54192	53312			1.65	1.65										
		3	52679	52679	51744			1.81	1.81										
		4	57328	57328	56448			1.56	1.56										
		5	57218	57218	56448			1.36	1.36										
55	**4**	1	32641	32665	31377			4.03	4.10			3.86	3.92	4.16	4.24				
		2	31154	31178	29918			4.13	4.21										
		3	30316	30340	29106			4.16	4.24										
		4	32798	32830	31542			3.98	4.08										
		5	32313	32313	31377			2.98	2.98										
55	**6**	1	26112	26082	20350			1.59	1.47			2.37	2.07	4.04	3.67				
		2	25784	25664	19914			3.58	3.09										
		3	25170	25080	19278			4.04	3.67										
		4	26328	26286	20483			1.57	1.41										
		5	26028	25938	20445			1.05	0.70										
55	**8**	1	25394	24458	23735			6.99	3.05			6.63	2.98	7.19	3.69				
		2	25320	24520	23818			6.31	2.95										
		3	24702	23894	23044			7.19	3.69										
		4	25565	24621	24044			6.33	2.40										
		5	25369	24529	23856			6.34	2.82										

Table D.8 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
10	**2**	1	2510	2510	2510	2510		0.00	0.00	0.00	0.00								
		2	2072	2072	2060	2072	892.25	0.58	0.58	0.00	0.00								
		3	1942	1942	1910	1942	1025.38	1.68	1.68	0.00	0.00	0.45	0.45	1.68	1.68	0.00	0.00	0.00	0.00
		4	2380	2380	2380	2380		0.00	0.00	0.00	0.00								
		5	2460	2460	2460	2460		0.00	0.00	0.00	0.00								
10	**4**	1	1902	1902	1550	1902	76.85	22.71	22.71	0.00	0.00								
		2	1648	1804	1352	1648	74.13	21.89	33.43	0.00	9.47								
		3	1598	1678	1280	1586	37.91	24.84	31.09	0.76	5.80	22.81	26.37	24.84	33.43	0.15	3.05	0.76	9.47
		4	1852	1852	1492	1852	35.51	24.13	24.13	0.00	0.00								
		5	1836	1836	1524	1836	19.91	20.47	20.47	0.00	0.00								
10	**6**	1	1972	1972	1770	1966	1.62	11.41	11.41	0.31	0.31								
		2	1950	1950	1628	1918	1.65	19.78	19.78	1.67	1.67								
		3	1878	1878	1544	1844	1.50	21.63	21.63	1.84	1.84	15.36	15.36	21.63	21.63	0.95	0.95	1.84	1.84
		4	1972	1972	1746	1960	1.21	12.94	12.94	0.61	0.61								
		5	1990	1990	1792	1984	1.05	11.05	11.05	0.30	0.30								
10	**8**	1	2470	2470	2144	2362	0.90	15.21	15.21	4.57	4.57								
		2	2476	2476	2136	2344	1.12	15.92	15.92	5.63	5.63								
		3	2382	2382	2040	2244	0.89	16.76	16.76	6.15	6.15	15.48	15.48	16.76	16.76	5.03	5.03	6.15	6.15
		4	2476	2476	2136	2356	0.58	15.92	15.92	5.09	5.09								
		5	2508	2508	2208	2418	0.33	13.59	13.59	3.72	3.72								

Table D.9: Computational results for *Fully Loaded Trips Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
25	**2**	1	13770	13770	13520			1.85	1.85			3.74	3.75	6.50	6.57				
		2	11008	11008	10478			5.06	5.06										
		3	10079	10086	9464			6.50	6.57										
		4	12881	12881	12506			3.00	3.00										
		5	13482	13482	13182			2.28	2.28										
25	**4**	1	8864	8864	8208			7.99	7.99			6.36	8.09	7.99	11.65				
		2	7382	7622	7032			4.98	8.39										
		3	6896	7208	6456			6.82	11.65										
		4	8386	8418	7824			7.18	7.59										
		5	8502	8502	8112			4.81	4.81										
25	**6**	1	7760	7736	7536			2.97	2.65			6.25	5.51	11.07	10.31				
		2	7490	7418	6888			8.74	7.69										
		3	7064	7016	6360			11.07	10.31										
		4	7682	7658	7344			4.60	4.28										
		5	7854	7758	7560			3.89	2.62										
25	**8**	1	8096	8024	7848			3.16	2.24			4.75	3.75	7.22	6.53				
		2	8006	7886	7512			6.58	4.98										
		3	7488	7440	6984			7.22	6.53										
		4	8058	8010	7776			3.63	3.01										
		5	8270	8174	8016			3.17	1.97										

Table D.9 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
40	**2**	1	34000	34000	34000			0.00	0.00			0.23	0.16	1.16	0.79				
		2	26800	26800	26800			0.00	0.00										
		3	24684	24592	24400			1.16	0.79										
		4	31680	31680	31680			0.00	0.00										
		5	33200	33200	33200			0.00	0.00										
40	**4**	1	20304	20288	20000			1.52	1.44			2.93	2.78	6.11	5.26				
		2	17080	17080	16480			3.64	3.64										
		3	16128	16000	15200			6.11	5.26										
		4	19600	19632	18960			3.38	3.54										
		5	19680	19680	19680			0.00	0.00										
40	**6**	1	16172	16052	14720			9.86	9.05			13.81	13.65	21.24	21.24				
		2	14716	14716	12468			18.03	18.03										
		3	14204	14204	11716			21.24	21.24										
		4	15684	15684	14076			11.42	11.42										
		5	15768	15768	14532			8.51	8.51										
40	**8**	1	16608	16464	16160			2.77	1.88			5.11	2.80	7.60	3.97				
		2	16328	15752	15224			7.25	3.47										
		3	15848	15312	14728			7.60	3.97										
		4	16648	16392	15880			4.84	3.22										
		5	16696	16440	16200			3.06	1.48										

Table D.9 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
55	**2**	1	63270	63270	62720			0.88	0.88			1.77	1.73	3.42	3.23				
		2	49653	49653	48608			2.15	2.15										
		3	45404	45322	43904			3.42	3.23										
		4	58841	58841	58016			1.42	1.42										
		5	61757	61757	61152			0.99	0.99										
55	**4**	1	36021	36005	34405			4.70	4.65			6.15	6.24	9.23	9.54				
		2	29629	29629	27647			7.17	7.17										
		3	27657	27737	25321			9.23	9.54										
		4	34445	34509	32299			6.64	6.84										
		5	34662	34662	33648			3.01	3.01										
55	**6**	1	28406	28328	27648			2.74	2.46			7.54	6.88	16.00	14.76				
		2	26276	25964	23436			12.12	10.79										
		3	25244	24974	21762			16.00	14.76										
		4	27566	27608	26406			4.39	4.55										
		5	27888	27714	27216			2.47	1.83										
55	**8**	1	26246	26054	24950			5.19	4.42			7.21	5.12	10.31	7.04				
		2	25347	24563	23365			8.48	5.13										
		3	24587	23859	22289			10.31	7.04										
		4	26004	25556	24291			7.05	5.21										
		5	25950	25654	24717			4.99	3.79										

Table D.9 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
10	**2**	1	1560	1560	1560	1560		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1440	1440	1440	1440		0.00	0.00	0.00	0.00								
		3	1380	1380	1380	1380		0.00	0.00	0.00	0.00								
		4	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
		5	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
10	**4**	1	966	966	936	958	9.01	3.21	3.21	0.84	0.84	11.57	11.57	19.68	19.68	0.70	0.70	2.27	2.27
		2	1034	1034	864	1030	33.52	19.68	19.68	0.39	0.39								
		3	900	900	828	880	15.40	8.70	8.70	2.27	2.27								
		4	1028	1028	936	1028	9.07	9.83	9.83	0.00	0.00								
		5	1090	1090	936	1090	11.11	16.45	16.45	0.00	0.00								
10	**6**	1	1030	1030	728	1030	0.90	41.48	41.48	0.00	0.00	50.31	50.31	63.16	63.16	0.90	0.90	4.49	4.49
		2	1178	1178	722	1178	1.21	63.16	63.16	0.00	0.00								
		3	994	994	644	994	1.80	54.35	54.35	0.00	0.00								
		4	1136	1136	808	1136	1.13	40.59	40.59	0.00	0.00								
		5	1304	1304	858	1248	1.01	51.98	51.98	4.49	4.49								
10	**8**	1	1428	1428	1056	1252	0.46	35.23	35.23	14.06	14.06	34.19	33.06	35.23	35.23	13.98	12.98	18.50	15.71
		2	1650	1650	1240	1426	0.52	33.06	33.06	15.71	15.71								
		3	1336	1336	992	1218	0.39	34.68	34.68	9.69	9.69								
		4	1478	1478	1096	1320	0.33	34.85	34.85	11.97	11.97								
		5	1704	1632	1280	1438	0.46	33.13	27.50	18.50	13.49								

Table D.10: Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
25	**2**	1	8788	8788	8788			0.00	0.00			0.00	0.00	0.00	0.00				
		2	8112	8112	8112			0.00	0.00										
		3	7774	7774	7774			0.00	0.00										
		4	8788	8788	8788			0.00	0.00										
		5	8788	8788	8788			0.00	0.00										
25	**4**	1	4760	4760	4732			0.59	0.59			1.51	1.38	5.54	5.33				
		2	4610	4601	4368			5.54	5.33										
		3	4214	4204	4186			0.67	0.43										
		4	4768	4758	4732			0.76	0.55										
		5	4732	4732	4732			0.00	0.00										
25	**6**	1	3660	3612	3380			8.28	6.86			14.14	12.60	20.92	21.25				
		2	4324	4336	3576			20.92	21.25										
		3	3445	3361	2990			15.22	12.41										
		4	3867	3757	3432			12.67	9.47										
		5	3954	3933	3480			13.62	13.02										
25	**8**	1	4282	4167	2856			49.93	45.90			38.52	35.99	52.79	47.90				
		2	5008	4872	3648			37.28	33.55										
		3	3997	3869	2616			52.79	47.90										
		4	4162	4162	3288			26.58	26.58										
		5	4537	4537	3600			26.03	26.03										

Table D.10 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
40	**2**	1	21840	21840	21840			0.00	0.00			0.00	0.00	0.00	0.00				
		2	20160	20160	20160			0.00	0.00										
		3	19320	19320	19320			0.00	0.00										
		4	21840	21840	21840			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
40	**4**	1	11488	11488	11440			0.42	0.42			0.55	0.45	1.06	0.71				
		2	10672	10632	10560			1.06	0.68										
		3	10200	10192	10120			0.79	0.71										
		4	11496	11488	11440			0.49	0.42										
		5	11440	11440	11440			0.00	0.00										
40	**6**	1	8116	8116	8008			1.35	1.35			5.66	4.87	8.93	8.50				
		2	8052	8020	7392			8.93	8.50										
		3	7674	7460	7084			8.33	5.31										
		4	8548	8540	8008			6.74	6.64										
		5	8244	8212	8008			2.95	2.55										
40	**8**	1	7680	7680	6280			22.29	22.29			25.27	24.86	39.59	39.05				
		2	8640	8656	6800			27.06	27.29										
		3	8208	8176	5880			39.59	39.05										
		4	8328	8240	7400			12.54	11.35										
		5	8792	8752	7040			24.89	24.32										

Table D.10 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
55	**2**	1	40768	40768	40768			0.00	0.00			0.00	0.00	0.00	0.00				
		2	37632	37632	37632			0.00	0.00										
		3	36064	36064	36064			0.00	0.00										
		4	40768	40768	40768			0.00	0.00										
		5	40768	40768	40768			0.00	0.00										
55	**4**	1	21216	21202	21112			0.49	0.43			0.44	0.41	0.85	0.78				
		2	19654	19640	19488			0.85	0.78										
		3	18758	18758	18676			0.44	0.44										
		4	21202	21202	21112			0.43	0.43										
		5	21112	21112	21112			0.00	0.00										
55	**6**	1	14851	14736	14560			2.00	1.21			3.06	2.61	6.28	6.01				
		2	14284	14248	13440			6.28	6.01										
		3	13324	13204	12880			3.45	2.52										
		4	15030	15004	14560			3.23	3.05										
		5	14612	14598	14560			0.36	0.26										
55	**8**	1	13032	13071	11284			15.49	15.84			16.39	16.03	21.73	21.73				
		2	13684	13684	11241			21.73	21.73										
		3	11893	11772	9982			19.14	17.93										
		4	13310	13195	11944			11.44	10.47										
		5	12881	12881	11284			14.15	14.15										

Table D.10 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
10	**2**	1	2160	2160	2160	2160		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	2040	2040	2040	2040		0.00	0.00	0.00	0.00								
		3	1980	1980	1980	1980		0.00	0.00	0.00	0.00								
		4	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
		5	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
10	**4**	1	1296	1296	1296	1296		0.00	0.00	0.00	0.00	1.05	1.05	2.31	2.31	0.00	0.00	0.00	0.00
		2	1236	1236	1224	1236	8.61	0.98	0.98	0.00	0.00								
		3	1200	1200	1188	1200	8.71	1.01	1.01	0.00	0.00								
		4	1308	1308	1296	1308	10.14	0.93	0.93	0.00	0.00								
		5	1326	1326	1296	1326	4.28	2.31	2.31	0.00	0.00								
10	**6**	1	1186	1186	1008	1186	0.97	17.66	17.66	0.00	0.00	27.90	27.90	37.61	37.61	0.59	0.59	2.63	2.63
		2	1310	1310	952	1310	1.07	37.61	37.61	0.00	0.00								
		3	1094	1094	924	1094	0.81	18.40	18.40	0.00	0.00								
		4	1316	1316	1008	1312	0.98	30.56	30.56	0.30	0.30								
		5	1404	1404	1038	1368	1.03	35.26	35.26	2.63	2.63								
10	**8**	1	1564	1528	1136	1412	0.33	37.68	34.51	10.76	8.22	36.70	32.75	37.69	37.69	12.34	8.95	19.90	14.51
		2	1786	1768	1320	1544	1.02	35.30	33.94	15.67	14.51								
		3	1476	1476	1072	1402	0.87	37.69	37.69	5.28	5.28								
		4	1614	1584	1176	1466	0.41	37.24	34.69	10.10	8.05								
		5	1844	1672	1360	1538	0.32	35.59	22.94	19.90	8.71								

Table D.11: Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
25	**2**	1	12168	12168	12168			0.00	0.00			0.00	0.00	0.00	0.00				
		2	11492	11492	11492			0.00	0.00										
		3	11154	11154	11154			0.00	0.00										
		4	12168	12168	12168			0.00	0.00										
		5	12168	12168	12168			0.00	0.00										
25	**4**	1	6562	6562	6552			0.15	0.15			0.21	0.21	0.58	0.58				
		2	6224	6224	6188			0.58	0.58										
		3	6016	6016	6006			0.17	0.17										
		4	6562	6562	6552			0.15	0.15										
		5	6552	6552	6552			0.00	0.00										
25	**6**	1	4720	4706	4680			0.85	0.56			3.91	3.79	11.04	11.04				
		2	5037	5037	4536			11.04	11.04										
		3	4386	4386	4290			2.24	2.24										
		4	4795	4781	4680			2.46	2.16										
		5	4819	4819	4680			2.97	2.97										
25	**8**	1	4580	4580	3744			22.33	22.33			19.02	18.95	24.33	23.98				
		2	5322	5322	4368			21.84	21.84										
		3	4267	4255	3432			24.33	23.98										
		4	4488	4488	4008			11.98	11.98										
		5	4951	4951	4320			14.61	14.61										

Table D.11 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
40	**2**	1	30240	30240	30240			0.00	0.00			0.00	0.00	0.00	0.00				
		2	28560	28560	28560			0.00	0.00										
		3	27720	27720	27720			0.00	0.00										
		4	30240	30240	30240			0.00	0.00										
		5	30240	30240	30240			0.00	0.00										
40	**4**	1	15848	15848	15840			0.05	0.05			0.09	0.09	0.21	0.21				
		2	14992	14992	14960			0.21	0.21										
		3	14536	14536	14520			0.11	0.11										
		4	15848	15848	15840			0.05	0.05										
		5	15840	15840	15840			0.00	0.00										
40	**6**	1	11140	11140	11088			0.47	0.47			0.70	0.52	1.60	0.92				
		2	10640	10568	10472			1.60	0.92										
		3	10248	10228	10164			0.83	0.63										
		4	11152	11152	11088			0.58	0.58										
		5	11088	11088	11088			0.00	0.00										
40	**8**	1	9032	8720	8640			4.54	0.93			5.70	3.52	9.09	7.68				
		2	9504	9224	8800			8.00	4.82										
		3	8640	8528	7920			9.09	7.68										
		4	9512	9512	9400			1.19	1.19										
		5	9552	9312	9040			5.66	3.01										

Table D.11 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
55	**2**	1	56448	56448	56448			0.00	0.00			0.00	0.00	0.00	0.00				
		2	53312	53312	53312			0.00	0.00										
		3	51744	51744	51744			0.00	0.00										
		4	56448	56448	56448			0.00	0.00										
		5	56448	56448	56448			0.00	0.00										
55	**4**	1	29254	29254	29232			0.08	0.08			0.07	0.07	0.16	0.16				
		2	27652	27652	27608			0.16	0.16										
		3	26810	26810	26796			0.05	0.05										
		4	29246	29246	29232			0.05	0.05										
		5	29232	29232	29232			0.00	0.00										
55	**6**	1	20286	20274	20160			0.63	0.57			0.55	0.47	1.08	0.79				
		2	19246	19190	19040			1.08	0.79										
		3	18570	18556	18480			0.49	0.41										
		4	20274	20274	20160			0.57	0.57										
		5	20160	20160	20160			0.00	0.00										
55	**8**	1	15808	15778	15624			1.18	0.99			2.76	2.44	5.91	5.16				
		2	15919	15807	15031			5.91	5.16										
		3	14659	14596	14322			2.35	1.91										
		4	16374	16342	15734			4.07	3.86										
		5	15670	15670	15624			0.29	0.29										

Table D.11 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
10	**2**	1	2400	2400	2400	2400		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860	1860		0.00	0.00	0.00	0.00								
		3	1680	1680	1680	1680		0.00	0.00	0.00	0.00								
		4	2220	2220	2220	2220		0.00	0.00	0.00	0.00								
		5	2340	2340	2340	2340		0.00	0.00	0.00	0.00								
10	**4**	1	1440	1440	1440	1440		0.00	0.00	0.00	0.00	2.24	3.60	6.45	13.26	0.00	1.28	0.00	6.40
		2	1188	1264	1116	1188	25.37	6.45	13.26	0.00	6.40								
		3	1056	1056	1008	1056	14.32	4.76	4.76	0.00	0.00								
		4	1332	1332	1332	1332		0.00	0.00	0.00	0.00								
		5	1404	1404	1404	1404		0.00	0.00	0.00	0.00								
10	**6**	1	1266	1266	1120	1266	0.92	13.04	13.04	0.00	0.00	31.33	31.33	46.54	46.54	0.67	0.67	1.99	1.99
		2	1272	1272	868	1272	1.34	46.54	46.54	0.00	0.00								
		3	1056	1056	784	1056	1.07	34.69	34.69	0.00	0.00								
		4	1358	1358	1036	1340	1.04	31.08	31.08	1.34	1.34								
		5	1434	1434	1092	1406	0.93	31.32	31.32	1.99	1.99								
10	**8**	1	1620	1496	1168	1480	0.78	38.70	28.08	9.46	1.08	37.07	31.31	38.70	37.40	11.85	6.98	20.28	15.30
		2	1756	1748	1296	1516	0.39	35.49	34.88	15.83	15.30								
		3	1418	1418	1032	1358	0.37	37.40	37.40	4.42	4.42								
		4	1628	1620	1184	1490	0.50	37.50	36.82	9.26	8.72								
		5	1886	1652	1384	1568	0.27	36.27	19.36	20.28	5.36								

Table D.12: Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
25	**2**	1	13520	13520	13520			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10478	10478	10478			0.00	0.00										
		3	9464	9464	9464			0.00	0.00										
		4	12506	12506	12506			0.00	0.00										
		5	13182	13182	13182			0.00	0.00										
25	**4**	1	7280	7280	7280			0.00	0.00			0.60	0.62	2.22	2.34				
		2	5767	5774	5642			2.22	2.34										
		3	5136	5136	5096			0.78	0.78										
		4	6734	6734	6734			0.00	0.00										
		5	7098	7098	7098			0.00	0.00										
25	**6**	1	5252	5252	5200			1.00	1.00			7.66	7.27	19.68	19.68				
		2	5084	5084	4248			19.68	19.68										
		3	4086	4058	3640			12.25	11.48										
		4	5020	4964	4810			4.37	3.20										
		5	5120	5120	5070			0.99	0.99										
25	**8**	1	4714	4714	4160			13.32	13.32			22.58	22.72	43.78	44.46				
		2	5365	5365	4152			29.21	29.21										
		3	4279	4299	2976			43.78	44.46										
		4	4606	4606	4080			12.89	12.89										
		5	5158	5158	4536			13.71	13.71										

Table D.12 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
40	**2**	1	33600	33600	33600			0.00	0.00			0.00	0.00	0.00	0.00				
		2	26040	26040	26040			0.00	0.00										
		3	23520	23520	23520			0.00	0.00										
		4	31080	31080	31080			0.00	0.00										
		5	32760	32760	32760			0.00	0.00										
40	**4**	1	17600	17600	17600			0.00	0.00			0.59	0.51	1.47	1.30				
		2	13840	13784	13640			1.47	1.06										
		3	12480	12480	12320			1.30	1.30										
		4	16312	16312	16280			0.20	0.20										
		5	17160	17160	17160			0.00	0.00										
40	**6**	1	12400	12384	12320			0.65	0.52			3.37	3.34	8.02	8.02				
		2	10112	10112	9548			5.91	5.91										
		3	9316	9316	8624			8.02	8.02										
		4	11652	11652	11396			2.25	2.25										
		5	12012	12012	12012			0.00	0.00										
40	**8**	1	9792	9744	9600			2.00	1.50			10.24	9.96	25.00	27.56				
		2	9544	9176	8200			16.39	11.90										
		3	8600	8776	6880			25.00	27.56										
		4	10048	10048	9600			4.67	4.67										
		5	9944	10040	9640			3.15	4.15										

Table D.12 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

n	K	Ins. #	FTH1	FTH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2	FTH1	FTH2
55	**2**	1	62720	62720	62720			0.00	0.00			0.00	0.00	0.00	0.00				
		2	48608	48608	48608			0.00	0.00										
		3	43904	43904	43904			0.00	0.00										
		4	58016	58016	58016			0.00	0.00										
		5	61152	61152	61152			0.00	0.00										
55	**4**	1	32480	32480	32480			0.00	0.00			0.33	0.31	0.79	0.77				
		2	25370	25352	25172			0.79	0.72										
		3	22912	22912	22736			0.77	0.77										
		4	30068	30068	30044			0.08	0.08										
		5	31668	31668	31668			0.00	0.00										
55	**6**	1	22580	22552	22400			0.80	0.68			2.35	2.21	5.00	4.25				
		2	18228	18092	17360			5.00	4.22										
		3	16198	16346	15680			3.30	4.25										
		4	21266	21120	20720			2.64	1.93										
		5	21840	21840	21840			0.00	0.00										
55	**8**	1	17698	17698	17360			1.95	1.95			7.40	6.29	13.21	11.66				
		2	15730	15514	13894			13.21	11.66										
		3	13712	13394	12152			12.84	10.22										
		4	17357	17197	16113			7.72	6.73										
		5	17145	17076	16926			1.29	0.89										

Table D.12 (continued): Computational results for *Fully Loaded Trips Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

Appendix E

Numerical Results for No Wait Problem

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
10	**2**	1	1560	1560	1560	1560		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1440	1440	1440	1440		0.00	0.00	0.00	0.00								
		3	1380	1380	1380	1380		0.00	0.00	0.00	0.00								
		4	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
		5	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
10	**4**	1	936	936	936	936		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	864	864	864	864		0.00	0.00	0.00	0.00								
		3	828	828	828	828		0.00	0.00	0.00	0.00								
		4	936	936	936	936		0.00	0.00	0.00	0.00								
		5	936	936	936	936		0.00	0.00	0.00	0.00								
10	**6**	1	728	728	728	728		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	672	672	672	672		0.00	0.00	0.00	0.00								
		3	644	644	644	644		0.00	0.00	0.00	0.00								
		4	728	728	728	728		0.00	0.00	0.00	0.00								
		5	780	780	780	780		0.00	0.00	0.00	0.00								
10	**8**	1	696	696	676	696	67.87	2.96	2.96	0.00	0.00	1.26	1.26	3.34	3.34	0.00	0.00	0.00	0.00
		2	672	672	672	672		0.00	0.00	0.00	0.00								
		3	618	618	598	618	92.17	3.34	3.34	0.00	0.00								
		4	728	728	728	728		0.00	0.00	0.00	0.00								
		5	780	780	780	780		0.00	0.00	0.00	0.00								

Table E.1: Computational results for *No Wait Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
25	**2**	1	8788	8788	8788			0.00	0.00			0.00	0.00	0.00	0.00				
		2	8112	8112	8112			0.00	0.00										
		3	7774	7774	7774			0.00	0.00										
		4	8788	8788	8788			0.00	0.00										
		5	8788	8788	8788			0.00	0.00										
25	**4**	1	4732	4732	4732			0.00	0.00			0.00	0.00	0.00	0.00				
		2	4368	4368	4368			0.00	0.00										
		3	4186	4186	4186			0.00	0.00										
		4	4732	4732	4732			0.00	0.00										
		5	4732	4732	4732			0.00	0.00										
25	**6**	1	3380	3380	3380			0.00	0.00			0.09	0.09	0.45	0.45				
		2	3134	3134	3120			0.45	0.45										
		3	2990	2990	2990			0.00	0.00										
		4	3380	3380	3380			0.00	0.00										
		5	3380	3380	3380			0.00	0.00										
25	**8**	1	2722	2722	2704			0.67	0.67			1.86	1.86	7.21	7.21				
		2	2676	2676	2496			7.21	7.21										
		3	2410	2410	2392			0.75	0.75										
		4	2722	2722	2704			0.67	0.67										
		5	2704	2704	2704			0.00	0.00										

Table E.1 (continued): Computational results for *No Wait Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
40	**2**	1	21840	21840	21840			0.00	0.00			0.00	0.00	0.00	0.00				
		2	20160	20160	20160			0.00	0.00										
		3	19320	19320	19320			0.00	0.00										
		4	21840	21840	21840			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
40	**4**	1	11440	11440	11440			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10560	10560	10560			0.00	0.00										
		3	10120	10120	10120			0.00	0.00										
		4	11440	11440	11440			0.00	0.00										
		5	11440	11440	11440			0.00	0.00										
40	**6**	1	8008	8008	8008			0.00	0.00			0.00	0.00	0.00	0.00				
		2	7392	7392	7392			0.00	0.00										
		3	7084	7084	7084			0.00	0.00										
		4	8008	8008	8008			0.00	0.00										
		5	8008	8008	8008			0.00	0.00										
40	**8**	1	6256	6256	6240			0.26	0.26			0.33	0.33	0.58	0.58				
		2	5792	5792	5760			0.56	0.56										
		3	5552	5552	5520			0.58	0.58										
		4	6256	6256	6240			0.26	0.26										
		5	6240	6240	6240			0.00	0.00										

Table E.1 (continued): Computational results for *No Wait Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
55	**2**	1	40768	40768	40768			0.00	0.00			0.00	0.00	0.00	0.00				
		2	37632	37632	37632			0.00	0.00										
		3	36064	36064	36064			0.00	0.00										
		4	40768	40768	40768			0.00	0.00										
		5	40768	40768	40768			0.00	0.00										
55	**4**	1	21112	21112	21112			0.00	0.00			0.00	0.00	0.00	0.00				
		2	19488	19488	19488			0.00	0.00										
		3	18676	18676	18676			0.00	0.00										
		4	21112	21112	21112			0.00	0.00										
		5	21112	21112	21112			0.00	0.00										
55	**6**	1	14560	14560	14560			0.00	0.00			0.00	0.00	0.00	0.00				
		2	13440	13440	13440			0.00	0.00										
		3	12880	12880	12880			0.00	0.00										
		4	14560	14560	14560			0.00	0.00										
		5	14560	14560	14560			0.00	0.00										
55	**8**	1	11298	11298	11284			0.12	0.12			0.14	0.14	0.29	0.29				
		2	10446	10446	10416			0.29	0.29										
		3	9996	9996	9982			0.14	0.14										
		4	11298	11298	11284			0.12	0.12										
		5	11284	11284	11284			0.00	0.00										

Table E.1 (continued): Computational results for *No Wait Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
10	**2**	1	2160	2160	2160	2160		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	2040	2040	2040	2040		0.00	0.00	0.00	0.00								
		3	1980	1980	1980	1980		0.00	0.00	0.00	0.00								
		4	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
		5	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
10	**4**	1	1296	1296	1296	1296		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1224	1224	1224	1224		0.00	0.00	0.00	0.00								
		3	1188	1188	1188	1188		0.00	0.00	0.00	0.00								
		4	1296	1296	1296	1296		0.00	0.00	0.00	0.00								
		5	1296	1296	1296	1296		0.00	0.00	0.00	0.00								
10	**6**	1	1008	1008	1008	1008		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	952	952	952	952		0.00	0.00	0.00	0.00								
		3	924	924	924	924		0.00	0.00	0.00	0.00								
		4	1008	1008	1008	1008		0.00	0.00	0.00	0.00								
		5	1008	1008	1008	1008		0.00	0.00	0.00	0.00								
10	**8**	1	864	864	864	864	51.64	0.00	0.00	0.00	0.00	1.67	1.67	8.33	8.33	0.00	0.00	0.00	0.00
		2	884	884	884	884		0.00	0.00	0.00	0.00								
		3	858	858	792	858		8.33	8.33	0.00	0.00								
		4	936	936	936	936		0.00	0.00	0.00	0.00								
		5	936	936	936	936		0.00	0.00	0.00	0.00								

Table E.2: Computational results for *No Wait Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
25	**2**	1	12168	12168	12168			0.00	0.00			0.00	0.00	0.00	0.00				
		2	11492	11492	11492			0.00	0.00										
		3	11154	11154	11154			0.00	0.00										
		4	12168	12168	12168			0.00	0.00										
		5	12168	12168	12168			0.00	0.00										
25	**4**	1	6552	6552	6552			0.00	0.00			0.00	0.00	0.00	0.00				
		2	6188	6188	6188			0.00	0.00										
		3	6006	6006	6006			0.00	0.00										
		4	6552	6552	6552			0.00	0.00										
		5	6552	6552	6552			0.00	0.00										
25	**6**	1	4680	4680	4680			0.00	0.00			0.00	0.00	0.00	0.00				
		2	4420	4420	4420			0.00	0.00										
		3	4290	4290	4290			0.00	0.00										
		4	4680	4680	4680			0.00	0.00										
		5	4680	4680	4680			0.00	0.00										
25	**8**	1	3744	3744	3744			0.00	0.00			0.10	0.10	0.51	0.51				
		2	3554	3554	3536			0.51	0.51										
		3	3432	3432	3432			0.00	0.00										
		4	3744	3744	3744			0.00	0.00										
		5	3744	3744	3744			0.00	0.00										

Table E.2 (continued): Computational results for *No Wait Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
40	**2**	1	30240	30240	30240			0.00	0.00			0.00	0.00	0.00	0.00				
		2	28560	28560	28560			0.00	0.00										
		3	27720	27720	27720			0.00	0.00										
		4	30240	30240	30240			0.00	0.00										
		5	30240	30240	30240			0.00	0.00										
40	**4**	1	15840	15840	15840			0.00	0.00			0.00	0.00	0.00	0.00				
		2	14960	14960	14960			0.00	0.00										
		3	14520	14520	14520			0.00	0.00										
		4	15840	15840	15840			0.00	0.00										
		5	15840	15840	15840			0.00	0.00										
40	**6**	1	11088	11088	11088			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10472	10472	10472			0.00	0.00										
		3	10164	10164	10164			0.00	0.00										
		4	11088	11088	11088			0.00	0.00										
		5	11088	11088	11088			0.00	0.00										
40	**8**	1	8640	8640	8640			0.00	0.00			0.00	0.00	0.00	0.00				
		2	8160	8160	8160			0.00	0.00										
		3	7920	7920	7920			0.00	0.00										
		4	8640	8640	8640			0.00	0.00										
		5	8640	8640	8640			0.00	0.00										

Table E.2 (continued): Computational results for *No Wait Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
55	**2**	1	56448	56448	56448			0.00	0.00			0.00	0.00	0.00	0.00				
		2	53312	53312	53312			0.00	0.00										
		3	51744	51744	51744			0.00	0.00										
		4	56448	56448	56448			0.00	0.00										
		5	56448	56448	56448			0.00	0.00										
55	**4**	1	29232	29232	29232			0.00	0.00			0.00	0.00	0.00	0.00				
		2	27608	27608	27608			0.00	0.00										
		3	26796	26796	26796			0.00	0.00										
		4	29232	29232	29232			0.00	0.00										
		5	29232	29232	29232			0.00	0.00										
55	**6**	1	20160	20160	20160			0.00	0.00			0.00	0.00	0.00	0.00				
		2	19040	19040	19040			0.00	0.00										
		3	18480	18480	18480			0.00	0.00										
		4	20160	20160	20160			0.00	0.00										
		5	20160	20160	20160			0.00	0.00										
55	**8**	1	15624	15624	15624			0.00	0.00			0.00	0.00	0.00	0.00				
		2	14756	14756	14756			0.00	0.00										
		3	14322	14322	14322			0.00	0.00										
		4	15624	15624	15624			0.00	0.00										
		5	15624	15624	15624			0.00	0.00										

Table E.2 (continued): Computational results for *No Wait Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
10	**2**	1	2400	2400	2400	2400		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860	1860		0.00	0.00	0.00	0.00								
		3	1680	1680	1680	1680		0.00	0.00	0.00	0.00								
		4	2220	2220	2220	2220		0.00	0.00	0.00	0.00								
		5	2340	2340	2340	2340		0.00	0.00	0.00	0.00								
10	**4**	1	1440	1440	1440	1440		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1116	1116	1116	1116		0.00	0.00	0.00	0.00								
		3	1008	1008	1008	1008		0.00	0.00	0.00	0.00								
		4	1332	1332	1332	1332		0.00	0.00	0.00	0.00								
		5	1404	1404	1404	1404		0.00	0.00	0.00	0.00								
10	**6**	1	1120	1120	1120	1120		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	868	868	868	868		0.00	0.00	0.00	0.00								
		3	784	784	784	784		0.00	0.00	0.00	0.00								
		4	1036	1036	1036	1036		0.00	0.00	0.00	0.00								
		5	1092	1092	1092	1092		0.00	0.00	0.00	0.00								
10	**8**	1	960	960	960	960	187.97 37.31 169.15	0.00	0.00	0.00	0.00	6.61	6.61	16.40	16.40	0.00	0.00	0.00	0.00
		2	866	866	744	866		16.40	16.40	0.00	0.00								
		3	728	728	672	728		8.33	8.33	0.00	0.00								
		4	962	962	888	962		8.33	8.33	0.00	0.00								
		5	1014	1014	1014	1014		0.00	0.00	0.00	0.00								

Table E.3: Computational results for *No Wait Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
25	**2**	1	13520	13520	13520			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10478	10478	10478			0.00	0.00										
		3	9464	9464	9464			0.00	0.00										
		4	12506	12506	12506			0.00	0.00										
		5	13182	13182	13182			0.00	0.00										
25	**4**	1	7280	7280	7280			0.00	0.00			0.00	0.00	0.00	0.00				
		2	5642	5642	5642			0.00	0.00										
		3	5096	5096	5096			0.00	0.00										
		4	6734	6734	6734			0.00	0.00										
		5	7098	7098	7098			0.00	0.00										
25	**6**	1	5200	5200	5200			0.00	0.00			0.00	0.00	0.00	0.00				
		2	4030	4030	4030			0.00	0.00										
		3	3640	3640	3640			0.00	0.00										
		4	4810	4810	4810			0.00	0.00										
		5	5070	5070	5070			0.00	0.00										
25	**8**	1	4160	4160	4160			0.00	0.00			0.33	0.33	1.67	1.67				
		2	3278	3278	3224			1.67	1.67										
		3	2912	2912	2912			0.00	0.00										
		4	3848	3848	3848			0.00	0.00										
		5	4056	4056	4056			0.00	0.00										

Table E.3 (continued): Computational results for *No Wait Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
40	**2**	1	33600	33600	33600			0.00	0.00										
		2	26040	26040	26040			0.00	0.00										
		3	23520	23520	23520			0.00	0.00			0.00	0.00	0.00	0.00				
		4	31080	31080	31080			0.00	0.00										
		5	32760	32760	32760			0.00	0.00										
40	**4**	1	17600	17600	17600			0.00	0.00										
		2	13640	13640	13640			0.00	0.00										
		3	12320	12320	12320			0.00	0.00			0.00	0.00	0.00	0.00				
		4	16280	16280	16280			0.00	0.00										
		5	17160	17160	17160			0.00	0.00										
40	**6**	1	12320	12320	12320			0.00	0.00										
		2	9548	9548	9548			0.00	0.00										
		3	8624	8624	8624			0.00	0.00			0.00	0.00	0.00	0.00				
		4	11396	11396	11396			0.00	0.00										
		5	12012	12012	12012			0.00	0.00										
40	**8**	1	9600	9600	9600			0.00	0.00										
		2	7440	7440	7440			0.00	0.00										
		3	6776	6776	6720			0.83	0.83			0.17	0.17	0.83	0.83				
		4	8880	8880	8880			0.00	0.00										
		5	9360	9360	9360			0.00	0.00										

Table E.3 (continued): Computational results for *No Wait Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
55	**2**	1	62720	62720	62720			0.00	0.00			0.00	0.00	0.00	0.00				
		2	48608	48608	48608			0.00	0.00										
		3	43904	43904	43904			0.00	0.00										
		4	58016	58016	58016			0.00	0.00										
		5	61152	61152	61152			0.00	0.00										
55	**4**	1	32480	32480	32480			0.00	0.00			0.00	0.00	0.00	0.00				
		2	25172	25172	25172			0.00	0.00										
		3	22736	22736	22736			0.00	0.00										
		4	30044	30044	30044			0.00	0.00										
		5	31668	31668	31668			0.00	0.00										
55	**6**	1	22400	22400	22400			0.00	0.00			0.00	0.00	0.00	0.00				
		2	17360	17360	17360			0.00	0.00										
		3	15680	15680	15680			0.00	0.00										
		4	20720	20720	20720			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
55	**8**	1	17360	17360	17360			0.00	0.00			0.00	0.00	0.00	0.00				
		2	13454	13454	13454			0.00	0.00										
		3	12152	12152	12152			0.00	0.00										
		4	16058	16058	16058			0.00	0.00										
		5	16926	16926	16926			0.00	0.00										

Table E.3 (continued): Computational results for *No Wait Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
10	**2**	1	1560	1560	1560	1560		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1440	1440	1440	1440		0.00	0.00	0.00	0.00								
		3	1380	1380	1380	1380		0.00	0.00	0.00	0.00								
		4	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
		5	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
10	**4**	1	936	936	936	936	7270.61	0.00	0.00	0.00	0.00	2.33	2.33	9.26	9.26	0.00	0.00	0.00	0.00
		2	944	944	864	944		9.26	9.26	0.00	0.00								
		3	848	848	828	848	2702.13	2.42	2.42	0.00	0.00								
		4	1040	1040	1040	1040		0.00	0.00	0.00	0.00								
		5	1040	1040	1040	1040		0.00	0.00	0.00	0.00								
10	**6**	1	844	844	832	844	433.45	1.44	1.44	0.00	0.00	4.46	4.46	10.42	10.42	0.00	0.00	0.00	0.00
		2	848	848	768	848	3067.61	10.42	10.42	0.00	0.00								
		3	802	802	736	802	1082.22	8.97	8.97	0.00	0.00								
		4	950	950	936	950	2251.22	1.50	1.50	0.00	0.00								
		5	936	936	936	936		0.00	0.00	0.00	0.00								
10	**8**	1	844	844	832	844	433.45	1.44	1.44	0.00	0.00	5.66	5.66	10.42	10.42	1.18	1.18	5.88	5.88
		2	848	848	768	848	4770.48	10.42	10.42	0.00	0.00								
		3	802	802	736	802	1252.86	8.97	8.97	0.00	0.00								
		4	898	898	884	898	2367.39	1.58	1.58	0.00	0.00								
		5	936	936	884	884	1513.15	5.88	5.88	5.88	5.88								

Table E.4: Computational results for *No Wait Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
25	**2**	1	8788	8788	8788			0.00	0.00			0.00	0.00	0.00	0.00				
		2	8112	8112	8112			0.00	0.00										
		3	7774	7774	7774			0.00	0.00										
		4	8788	8788	8788			0.00	0.00										
		5	8788	8788	8788			0.00	0.00										
25	**4**	1	4732	4732	4732			0.00	0.00			0.92	0.92	4.58	4.58				
		2	4568	4568	4368			4.58	4.58										
		3	4186	4186	4186			0.00	0.00										
		4	4732	4732	4732			0.00	0.00										
		5	4732	4732	4732			0.00	0.00										
25	**6**	1	3814	3814	3796			0.47	0.47			2.78	2.85	6.74	6.74				
		2	3722	3734	3504			6.22	6.56										
		3	3388	3388	3174			6.74	6.74										
		4	3814	3814	3796			0.47	0.47										
		5	3796	3796	3796			0.00	0.00										
25	**8**	1	3380	3370	3328			1.56	1.26			3.39	3.33	7.36	7.36				
		2	3298	3298	3072			7.36	7.36										
		3	2996	2996	2806			6.77	6.77										
		4	3370	3370	3328			1.26	1.26										
		5	3328	3328	3328			0.00	0.00										

Table E.4 (continued): Computational results for *No Wait Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
40	**2**	1	21840	21840	21840			0.00	0.00			0.00	0.00	0.00	0.00				
		2	20160	20160	20160			0.00	0.00										
		3	19320	19320	19320			0.00	0.00										
		4	21840	21840	21840			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
40	**4**	1	11440	11440	11440			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10560	10560	10560			0.00	0.00										
		3	10120	10120	10120			0.00	0.00										
		4	11440	11440	11440			0.00	0.00										
		5	11440	11440	11440			0.00	0.00										
40	**6**	1	8354	8354	8320			0.41	0.41			2.29	2.27	6.03	5.90				
		2	8034	8034	7680			4.61	4.61										
		3	7804	7794	7360			6.03	5.90										
		4	8668	8668	8632			0.42	0.42										
		5	8632	8632	8632			0.00	0.00										
40	**8**	1	7128	7112	7020			1.54	1.31			3.33	3.33	7.67	7.67				
		2	6876	6876	6480			6.11	6.11										
		3	6686	6686	6210			7.67	7.67										
		4	7376	7392	7280			1.32	1.54										
		5	7280	7280	7280			0.00	0.00										

Table E.4 (continued): Computational results for *No Wait Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
55	**2**	1	40768	40768	40768			0.00	0.00			0.00	0.00	0.00	0.00				
		2	37632	37632	37632			0.00	0.00										
		3	36064	36064	36064			0.00	0.00										
		4	40768	40768	40768			0.00	0.00										
		5	40768	40768	40768			0.00	0.00										
55	**4**	1	21112	21112	21112			0.00	0.00			0.00	0.00	0.00	0.00				
		2	19488	19488	19488			0.00	0.00										
		3	18676	18676	18676			0.00	0.00										
		4	21112	21112	21112			0.00	0.00										
		5	21112	21112	21112			0.00	0.00										
55	**6**	1	15096	15096	15028			0.45	0.45			1.78	1.77	3.75	3.75				
		2	14368	14392	13872			3.58	3.75										
		3	13792	13780	13294			3.75	3.66										
		4	15556	15538	15496			0.39	0.27										
		5	15138	15138	15028			0.73	0.73										
55	**8**	1	12480	12460	12324			1.27	1.10			2.57	2.52	4.96	5.06				
		2	11940	11952	11376			4.96	5.06										
		3	11386	11386	10902			4.44	4.44										
		4	12850	12828	12688			1.28	1.10										
		5	12434	12434	12324			0.89	0.89										

Table E.4 (continued): Computational results for *No Wait Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
10	**2**	1	2160	2160	2160	2160		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	2040	2040	2040	2040		0.00	0.00	0.00	0.00								
		3	1980	1980	1980	1980		0.00	0.00	0.00	0.00								
		4	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
		5	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
10	**4**	1	1296	1296	1296	1296		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1224	1224	1224	1224		0.00	0.00	0.00	0.00								
		3	1188	1188	1188	1188		0.00	0.00	0.00	0.00								
		4	1296	1296	1296	1296		0.00	0.00	0.00	0.00								
		5	1296	1296	1296	1296		0.00	0.00	0.00	0.00								
10	**6**	1	1080	1080	1080	1080		0.00	0.00	0.00	0.00	1.94	1.94	7.84	7.84	0.00	0.00	0.00	0.00
		2	1100	1100	1020	1100	857.61	7.84	7.84	0.00	0.00								
		3	990	990	990	990		0.00	0.00	0.00	0.00								
		4	1100	1100	1080	1100	466.39	1.85	1.85	0.00	0.00								
		5	1152	1152	1152	1152		0.00	0.00	0.00	0.00								
10	**8**	1	1080	1080	1080	1080		0.00	0.00	0.00	0.00	1.94	1.94	7.84	7.84	0.00	0.00	0.00	0.00
		2	1100	1100	1020	1100	857.61	7.84	7.84	0.00	0.00								
		3	990	990	990	990		0.00	0.00	0.00	0.00								
		4	1100	1100	1080	1100	466.39	1.85	1.85	0.00	0.00								
		5	1152	1152	1152	1152		0.00	0.00	0.00	0.00								

Table E.5: Computational results for *No Wait Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
25	**2**	1	12168	12168	12168			0.00	0.00			0.00	0.00	0.00	0.00				
		2	11492	11492	11492			0.00	0.00										
		3	11154	11154	11154			0.00	0.00										
		4	12168	12168	12168			0.00	0.00										
		5	12168	12168	12168			0.00	0.00										
25	**4**	1	6552	6552	6552			0.00	0.00			0.00	0.00	0.00	0.00				
		2	6188	6188	6188			0.00	0.00										
		3	6006	6006	6006			0.00	0.00										
		4	6552	6552	6552			0.00	0.00										
		5	6552	6552	6552			0.00	0.00										
25	**6**	1	4680	4680	4680			0.00	0.00			0.21	0.21	1.07	1.07				
		2	4742	4742	4692			1.07	1.07										
		3	4290	4290	4290			0.00	0.00										
		4	4968	4968	4968			0.00	0.00										
		5	4968	4968	4968			0.00	0.00										
25	**8**	1	4198	4198	4176			0.53	0.53			0.69	0.69	1.78	1.78				
		2	4222	4222	4148			1.78	1.78										
		3	3850	3850	3828			0.57	0.57										
		4	4416	4416	4392			0.55	0.55										
		5	4392	4392	4392			0.00	0.00										

Table E.5 (continued): Computational results for *No Wait Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
40	**2**	1	30240	30240	30240			0.00	0.00			0.00	0.00	0.00	0.00				
		2	28560	28560	28560			0.00	0.00										
		3	27720	27720	27720			0.00	0.00										
		4	30240	30240	30240			0.00	0.00										
		5	30240	30240	30240			0.00	0.00										
40	**4**	1	15840	15840	15840			0.00	0.00			0.00	0.00	0.00	0.00				
		2	14960	14960	14960			0.00	0.00										
		3	14520	14520	14520			0.00	0.00										
		4	15840	15840	15840			0.00	0.00										
		5	15840	15840	15840			0.00	0.00										
40	**6**	1	11088	11088	11088			0.00	0.00			0.15	0.15	0.76	0.76				
		2	10552	10552	10472			0.76	0.76										
		3	10164	10164	10164			0.00	0.00										
		4	11520	11520	11520			0.00	0.00										
		5	11088	11088	11088			0.00	0.00										
40	**8**	1	9396	9396	9360			0.38	0.38			1.40	1.44	5.07	5.26				
		2	8956	8956	8840			1.31	1.31										
		3	8668	8684	8250			5.07	5.26										
		4	9742	9742	9720			0.23	0.23										
		5	9360	9360	9360			0.00	0.00										

Table E.5 (continued): Computational results for *No Wait Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
55	**2**	1	56448	56448	56448			0.00	0.00			0.00	0.00	0.00	0.00				
		2	53312	53312	53312			0.00	0.00										
		3	51744	51744	51744			0.00	0.00										
		4	56448	56448	56448			0.00	0.00										
		5	56448	56448	56448			0.00	0.00										
55	**4**	1	29232	29232	29232			0.00	0.00			0.00	0.00	0.00	0.00				
		2	27608	27608	27608			0.00	0.00										
		3	26796	26796	26796			0.00	0.00										
		4	29232	29232	29232			0.00	0.00										
		5	29232	29232	29232			0.00	0.00										
55	**6**	1	20160	20160	20160			0.00	0.00			0.12	0.12	0.58	0.58				
		2	19150	19150	19040			0.58	0.58										
		3	18480	18480	18480			0.00	0.00										
		4	20160	20160	20160			0.00	0.00										
		5	20160	20160	20160			0.00	0.00										
55	**8**	1	16610	16610	16560			0.30	0.30			1.03	1.02	3.60	3.48				
		2	15784	15800	15640			0.92	1.02										
		3	15248	15230	14718			3.60	3.48										
		4	16610	16610	16560			0.30	0.30										
		5	16560	16560	16560			0.00	0.00										

Table E.5 (continued): Computational results for *No Wait Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
10	**2**	1	2400	2400	2400	2400		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860	1860		0.00	0.00	0.00	0.00								
		3	1680	1680	1680	1680		0.00	0.00	0.00	0.00								
		4	2220	2220	2220	2220		0.00	0.00	0.00	0.00								
		5	2340	2340	2340	2340		0.00	0.00	0.00	0.00								
10	**4**	1	1440	1440	1440	1440		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1116	1116	1116	1116		0.00	0.00	0.00	0.00								
		3	1008	1008	1008	1008		0.00	0.00	0.00	0.00								
		4	1332	1332	1332	1332		0.00	0.00	0.00	0.00								
		5	1404	1404	1404	1404		0.00	0.00	0.00	0.00								
10	**6**	1	1180	1180	1120	1180	342.91	5.36	5.36	0.00	0.00	8.36	8.36	17.35	17.35	0.35	0.35	1.76	1.76
		2	1028	1028	930	1028	2477.41	10.54	10.54	0.00	0.00								
		3	920	920	784	920	412.89	17.35	17.35	0.00	0.00								
		4	1110	1110	1110	1110		0.00	0.00	0.00	0.00								
		5	1270	1270	1170	1248	2364.57	8.55	8.55	1.76	1.76								
10	**8**	1	1180	1180	1120	1180	342.91	5.36	5.36	0.00	0.00	8.36	8.36	17.35	17.35	0.35	0.35	1.76	1.76
		2	1028	1028	930	1028	3213.26	10.54	10.54	0.00	0.00								
		3	920	920	784	920	310.35	17.35	17.35	0.00	0.00								
		4	1110	1110	1110	1110		0.00	0.00	0.00	0.00								
		5	1270	1270	1170	1248	2364.57	8.55	8.55	1.76	1.76								

Table E.6: Computational results for *No Wait Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
25	**2**	1	13520	13520	13520			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10478	10478	10478			0.00	0.00										
		3	9464	9464	9464			0.00	0.00										
		4	12506	12506	12506			0.00	0.00										
		5	13182	13182	13182			0.00	0.00										
25	**4**	1	7280	7280	7280			0.00	0.00			0.00	0.00	0.00	0.00				
		2	5642	5642	5642			0.00	0.00										
		3	5096	5096	5096			0.00	0.00										
		4	6734	6734	6734			0.00	0.00										
		5	7098	7098	7098			0.00	0.00										
25	**6**	1	5200	5200	5200			0.00	0.00			5.14	5.14	13.65	13.65				
		2	4580	4580	4030			13.65	13.65										
		3	3928	3928	3640			7.91	7.91										
		4	5010	5010	4810			4.16	4.16										
		5	5070	5070	5070			0.00	0.00										
25	**8**	1	4440	4440	4400			0.91	0.91			8.44	8.44	22.80	22.80				
		2	4094	4094	3596			13.85	13.85										
		3	3576	3576	2912			22.80	22.80										
		4	4492	4492	4292			4.66	4.66										
		5	4524	4524	4524			0.00	0.00										

Table E.6 (continued): Computational results for *No Wait Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
40	**2**	1	33600	33600	33600			0.00	0.00			0.00	0.00	0.00	0.00				
		2	26040	26040	26040			0.00	0.00										
		3	23520	23520	23520			0.00	0.00										
		4	31080	31080	31080			0.00	0.00										
		5	32760	32760	32760			0.00	0.00										
40	**4**	1	17600	17600	17600			0.00	0.00			0.00	0.00	0.00	0.00				
		2	13640	13640	13640			0.00	0.00										
		3	12320	12320	12320			0.00	0.00										
		4	16280	16280	16280			0.00	0.00										
		5	17160	17160	17160			0.00	0.00										
40	**6**	1	12320	12320	12320			0.00	0.00			2.41	2.41	4.64	4.64				
		2	9986	9986	9548			4.59	4.59										
		3	9024	9024	8624			4.64	4.64										
		4	11716	11716	11396			2.81	2.81										
		5	12012	12012	12012			0.00	0.00										
40	**8**	1	10036	10036	10000			0.36	0.36			6.79	6.81	15.24	16.19				
		2	8598	8532	7750			10.94	10.09										
		3	7744	7808	6720			15.24	16.19										
		4	9940	9940	9620			3.33	3.33										
		5	10150	10150	9750			4.10	4.10										

Table E.6 (continued): Computational results for *No Wait Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
55	**2**	1	62720	62720	62720			0.00	0.00			0.00	0.00	0.00	0.00				
		2	48608	48608	48608			0.00	0.00										
		3	43904	43904	43904			0.00	0.00										
		4	58016	58016	58016			0.00	0.00										
		5	61152	61152	61152			0.00	0.00										
55	**4**	1	32480	32480	32480			0.00	0.00			0.00	0.00	0.00	0.00				
		2	25172	25172	25172			0.00	0.00										
		3	22736	22736	22736			0.00	0.00										
		4	30044	30044	30044			0.00	0.00										
		5	31668	31668	31668			0.00	0.00										
55	**6**	1	22400	22400	22400			0.00	0.00			1.33	1.33	3.49	3.49				
		2	17966	17966	17360			3.49	3.49										
		3	16176	16176	15680			3.16	3.16										
		4	20720	20720	20720			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
55	**8**	1	17904	17904	17840			0.36	0.36			4.31	4.24	11.85	11.85				
		2	15006	14958	13826			8.53	8.19										
		3	13592	13592	12152			11.85	11.85										
		4	17156	17156	17020			0.80	0.80										
		5	17394	17394	17394			0.00	0.00										

Table E.6 (continued): Computational results for *No Wait Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
10	**2**	1	1834	1834	1820	1716(*)		0.77	0.77			2.03	2.12	6.58	6.83				
		2	1710	1714	1680			1.79	2.02										
		3	1716	1720	1610			6.58	6.83										
		4	1838	1838	1820			0.99	0.99										
		5	1820	1820	1820			0.00	0.00										
10	**4**	1	1522	1434	1404	1434	34608.51	8.40	2.14	6.14	0.00	10.34	9.09	16.20	16.20				
		2	1506	1506	1296	1388	103433.02	16.20	16.20	8.50	8.50								
		3	1354	1354	1242	1354	127726.17	9.02	9.02	0.00	0.00								
		4	1540	1540	1404	1446	34851.46	9.69	9.69	6.50	6.50								
		5	1522	1522	1404			8.40	8.40	---	---								
10	**6**	1	1522	1434	1352	1434	34608.51	12.57	6.07	6.14	0.00	11.18	9.88	16.20	16.20				
		2	1506	1506	1296	1388	103433.02	16.20	16.20	8.50	8.50								
		3	1354	1354	1242	1354	127726.17	9.02	9.02	0.00	0.00								
		4	1540	1540	1404	1446	34851.46	9.69	9.69	6.50	6.50								
		5	1522	1522	1404			8.40	8.40	---	---								
10	**8**	1	1522	1434	1352	1434	34608.51	12.57	6.07	6.14	0.00	11.18	9.88	16.20	16.20				
		2	1506	1506	1296	1388	103433.02	16.20	16.20	8.50	8.50								
		3	1354	1354	1242	1354	127726.17	9.02	9.02	0.00	0.00								
		4	1540	1540	1404	1446	34851.46	9.69	9.69	6.50	6.50								
		5	1522	1522	1404	1522(*)		8.40	8.40										

Table E.7: Computational results for *No Wait Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
25	**2**	1	9460	9464	9412			0.51	0.55			1.21	1.30	3.63	3.77				
		2	8802	8818	8688			1.31	1.50										
		3	8628	8640	8326			3.63	3.77										
		4	9468	9476	9412			0.59	0.68										
		5	9412	9412	9412			0.00	0.00										
25	**4**	1	6310	6310	5980			5.52	5.52			7.47	6.39	16.30	14.53				
		2	6420	6322	5520			16.30	14.53										
		3	5820	5770	5290			10.02	9.07										
		4	6106	6106	5980			2.11	2.11										
		5	6184	6022	5980			3.41	0.70										
25	**6**	1	5584	5532	5200			7.38	6.38			13.73	11.72	20.71	18.35				
		2	5968	5758	4944			20.71	16.46										
		3	5520	5444	4600			20.00	18.35										
		4	5744	5744	5200			10.46	10.46										
		5	5838	5672	5304			10.07	6.94										
25	**8**	1	5584	5532	5096			9.58	8.56			15.80	13.76	23.10	20.76				
		2	5968	5758	4848			23.10	18.77										
		3	5520	5444	4508			22.45	20.76										
		4	5744	5744	5096			12.72	12.72										
		5	5838	5672	5252			11.16	8.00										

Table E.7 (continued): Computational results for *No Wait Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
40	**2**	1	22974	22990	22880			0.41	0.48			0.92	1.06	2.67	2.92				
		2	21332	21380	21120			1.00	1.23										
		3	20780	20832	20240			2.67	2.92										
		4	22994	23034	22880			0.50	0.67										
		5	22880	22880	22880			0.00	0.00										
40	**4**	1	13746	13738	13468			2.06	2.00			4.09	3.96	8.39	8.39				
		2	13184	13240	12432			6.05	6.50										
		3	12914	12914	11914			8.39	8.39										
		4	13754	13754	13468			2.12	2.12										
		5	13714	13572	13468			1.83	0.77										
40	**6**	1	12134	11814	10972			10.59	7.67			15.56	14.19	23.26	19.35				
		2	12136	12088	10128			19.83	19.35										
		3	11964	11578	9706			23.26	19.29										
		4	12294	12348	10972			12.05	12.54										
		5	12298	12298	10972			12.09	12.09										
40	**8**	1	12134	11814	10296			17.85	14.74			24.22	22.73	34.07	29.81				
		2	12136	12088	9312			30.33	29.81										
		3	11964	11578	8924			34.07	29.74										
		4	12294	12348	10296			19.41	19.93										
		5	12298	12298	10296			19.44	19.44										

Table E.7 (continued): Computational results for *No Wait Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
55	**2**	1	42352	42396	42172			0.43	0.53			0.79	0.95	2.15	2.44				
		2	39290	39374	38928			0.93	1.15										
		3	38108	38216	37306			2.15	2.44										
		4	42364	42436	42172			0.46	0.63										
		5	42172	42172	42172			0.00	0.00										
55	**4**	1	24414	24884	23920			2.07	4.03			4.58	5.02	7.85	7.55				
		2	23300	23412	22080			5.53	6.03										
		3	22822	22758	21160			7.85	7.55										
		4	24998	24998	23920			4.51	4.51										
		5	24620	24628	23920			2.93	2.96										
55	**6**	1	20872	20552	18720			11.50	9.79			15.42	14.78	22.69	20.88				
		2	20770	20876	17280			20.20	20.81										
		3	20318	20018	16560			22.69	20.88										
		4	20872	20872	18720			11.50	11.50										
		5	20820	20768	18720			11.22	10.94										
55	**8**	1	20872	20552	16952			23.12	21.24			28.29	27.59	37.60	35.57				
		2	20770	20876	15408			34.80	35.49										
		3	20318	20018	14766			37.60	35.57										
		4	20872	20872	16952			23.12	23.12										
		5	20820	20768	16952			22.82	22.51										

Table E.7 (continued): Computational results for *No Wait Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
10	**2**	1	2520	2520	2520			0.00	0.00			0.00	0.00	0.00	0.00				
		2	2380	2380	2380			0.00	0.00										
		3	2310	2310	2310			0.00	0.00										
		4	2520	2520	2520			0.00	0.00										
		5	2520	2520	2520			0.00	0.00										
10	**4**	1	1818	1818	1800			1.00	1.00			7.16	5.63	9.09	9.09				
		2	1854	1854	1700	1790(*)		9.06	9.06										
		3	1800	1800	1650	1740	104625.72	9.09	9.09	3.45	3.45								
		4	1956	1818	1800			8.67	1.00										
		5	1944	1944	1800	1818	55740.93	8.00	8.00	6.93	6.93								
10	**6**	1	1756	1756	1728	1756	27679.80	1.62	1.62	0.00	0.00	9.06	5.08	13.26	9.09	5.65	1.79	7.90	3.78
		2	1834	1766	1632	1722	59274.44	12.38	8.21	6.50	2.56								
		3	1794	1728	1584	1684	70499.34	13.26	9.09	6.53	2.61								
		4	1884	1756	1728	1756	64088.75	9.03	1.62	7.29	0.00								
		5	1884	1812	1728	1746	28243.47	9.03	4.86	7.90	3.78								
10	**8**	1	1756	1756	1728	1756	27679.80	1.62	1.62	0.00	0.00	9.06	5.08	13.26	9.09	5.65	1.79	7.90	3.78
		2	1834	1766	1632	1722	59274.44	12.38	8.21	6.50	2.56								
		3	1794	1728	1584	1684	70499.34	13.26	9.09	6.53	2.61								
		4	1884	1756	1728	1756	64088.75	9.03	1.62	7.29	0.00								
		5	1884	1812	1728	1746	28243.47	9.03	4.86	7.90	3.78								

Table E.8: Computational results for *No Wait Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
25	**2**	1	13032	13032	13032			0.00	0.00			0.00	0.00	0.00	0.00				
		2	12308	12308	12308			0.00	0.00										
		3	11946	11946	11946			0.00	0.00										
		4	13032	13032	13032			0.00	0.00										
		5	13032	13032	13032			0.00	0.00										
25	**4**	1	7928	7936	7848			1.02	1.12			3.16	3.20	6.89	6.89				
		2	7920	7920	7412			6.85	6.85										
		3	7690	7690	7194			6.89	6.89										
		4	7928	7936	7848			1.02	1.12										
		5	7848	7848	7848			0.00	0.00										
25	**6**	1	6800	6800	6696			1.55	1.55			4.10	4.10	8.80	8.80				
		2	6830	6830	6324			8.00	8.00										
		3	6678	6678	6138			8.80	8.80										
		4	6800	6800	6696			1.55	1.55										
		5	6736	6736	6696			0.60	0.60										
25	**8**	1	6654	6436	6264			6.23	2.75			7.44	5.38	12.86	10.58				
		2	6830	6692	6052			12.86	10.58										
		3	6544	6410	5874			11.41	9.12										
		4	6654	6508	6408			3.84	1.56										
		5	6592	6592	6408			2.87	2.87										

Table E.8 (continued): Computational results for *No Wait Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
40	**2**	1	31680	31680	31680			0.00	0.00			0.00	0.00	0.00	0.00				
		2	29920	29920	29920			0.00	0.00										
		3	29040	29040	29040			0.00	0.00										
		4	31680	31680	31680			0.00	0.00										
		5	31680	31680	31680			0.00	0.00										
40	**4**	1	18184	18208	18000			1.02	1.16			1.63	1.00	4.90	1.39				
		2	17206	17222	17000			1.21	1.31										
		3	17308	16730	16500			4.90	1.39										
		4	18184	18208	18000			1.02	1.16										
		5	18000	18000	18000			0.00	0.00										
40	**6**	1	14836	14544	14328			3.55	1.51			3.79	2.56	8.18	4.98				
		2	14082	14082	13532			4.06	4.06										
		3	14208	13788	13134			8.18	4.98										
		4	14628	14544	14328			2.09	1.51										
		5	14482	14432	14328			1.07	0.73										
40	**8**	1	14388	13390	12816			12.27	4.48			9.29	6.23	14.22	10.72				
		2	13798	13442	12376			11.49	8.61										
		3	13720	13300	12012			14.22	10.72										
		4	13610	13538	13104			3.86	3.31										
		5	13708	13634	13104			4.61	4.04										

Table E.8 (continued): Computational results for *No Wait Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
55	**2**	1	58392	58392	58392			0.00	0.00			0.00	0.00	0.00	0.00				
		2	55148	55148	55148			0.00	0.00										
		3	53526	53526	53526			0.00	0.00										
		4	58392	58392	58392			0.00	0.00										
		5	58392	58392	58392			0.00	0.00										
55	**4**	1	32520	32560	32184			1.04	1.17			0.92	1.01	1.27	1.38				
		2	30772	30796	30396			1.24	1.32										
		3	29878	29910	29502			1.27	1.38										
		4	32520	32560	32184			1.04	1.17										
		5	32184	32184	32184			0.00	0.00										
55	**6**	1	25588	25088	24696			3.61	1.59			3.18	2.45	5.42	4.68				
		2	24418	24220	23324			4.69	3.84										
		3	23866	23698	22638			5.42	4.68										
		4	25088	25088	24696			1.59	1.59										
		5	24836	24836	24696			0.57	0.57										
55	**8**	1	22992	22698	21384			7.52	6.14			9.42	8.17	13.37	12.29				
		2	22898	22754	20604			11.13	10.43										
		3	22222	22012	19602			13.37	12.29										
		4	22992	22552	21384			7.52	5.46										
		5	23000	22782	21384			7.56	6.54										

Table E.8 (continued): Computational results for *No Wait Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
10	**2**	1	2800	2800	2800	2800		0.00	0.00										
		2	2170	2170	2170	2170		0.00	0.00			0.24	0.24	1.22	1.22				
		3	1984	1984	1960			1.22	1.22										
		4	2590	2590	2590	2590		0.00	0.00										
		5	2730	2730	2730	2730		0.00	0.00										
10	**4**	1	2036	2036	2000			1.80	1.80										
		2	1740	1728	1550	1728	61045.96	12.26	11.48	0.69	0.00	6.45	6.29	14.29	14.29				
		3	1600	1600	1400	1600	88992.59	14.29	14.29	0.00	0.00								
		4	1922	1922	1850			3.89	3.89										
		5	1950	1950	1950	1950		0.00	0.00	0.00	0.00								
10	**6**	1	1956	1956	1920	1956	39908.59	1.88	1.88	0.00	0.00								
		2	1846	1846	1488	1728(*)		24.06	24.06	---	---	9.81	9.81	24.06	24.06				
		3	1600	1600	1344	1600	88992.59	19.05	19.05	0.00	0.00								
		4	1848	1848	1776	1848	88297.99	4.05	4.05	0.00	0.00								
		5	1872	1872	1872	1872		0.00	0.00	0.00	0.00								
10	**8**	1	1956	1956	1920	1956	39908.59	1.88	1.88	0.00	0.00								
		2	1846	1846	1488	1728(*)		24.06	24.06	---	---	9.81	9.81	24.06	24.06				
		3	1600	1600	1344	1600	88992.59	19.05	19.05	0.00	0.00								
		4	1848	1848	1776	1848	88297.99	4.05	4.05	0.00	0.00								
		5	1872	1872	1872	1872		0.00	0.00	0.00	0.00								

Table E.9: Computational results for *No Wait Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
25	**2**	1	14480	14480	14480			0.00	0.00			0.12	0.12	0.47	0.47				
		2	11234	11234	11222			0.11	0.11										
		3	10184	10184	10136			0.47	0.47										
		4	13394	13394	13394			0.00	0.00										
		5	14118	14118	14118			0.00	0.00										
25	**4**	1	8864	8864	8720			1.65	1.65			5.52	5.60	12.06	12.06				
		2	7430	7430	6758			9.94	9.94										
		3	6840	6840	6104			12.06	12.06										
		4	8386	8418	8066			3.97	4.36										
		5	8502	8502	8502			0.00	0.00										
25	**6**	1	7806	7640	7120			9.63	7.30			12.67	12.33	24.69	24.69				
		2	6842	6842	5766			18.66	18.66										
		3	6494	6494	5208			24.69	24.69										
		4	7406	7406	6882			7.61	7.61										
		5	7454	7500	7254			2.76	3.39										
25	**8**	1	7196	7032	6560			9.70	7.20			15.26	14.30	30.83	30.83				
		2	6712	6712	5394			24.43	24.43										
		3	6374	6374	4872			30.83	30.83										
		4	6830	6830	6438			6.09	6.09										
		5	7142	6986	6786			5.25	2.95										

Table E.9 (continued): Computational results for *No Wait Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
40	**2**	1	35200	35200	35200			0.00	0.00			0.12	0.12	0.58	0.58				
		2	27280	27280	27280			0.00	0.00										
		3	24784	24784	24640			0.58	0.58										
		4	32560	32560	32560			0.00	0.00										
		5	34320	34320	34320			0.00	0.00										
40	**4**	1	20292	20324	20000			1.46	1.62			5.01	4.96	11.60	10.80				
		2	16772	16772	15500			8.21	8.21										
		3	15624	15512	14000			11.60	10.80										
		4	19204	19268	18500			3.81	4.15										
		5	19500	19500	19500			0.00	0.00										
40	**6**	1	16492	16492	15440			6.81	6.81			13.49	13.57	29.90	30.31				
		2	14744	14744	12338			19.50	19.50										
		3	14040	14084	10808			29.90	30.31										
		4	15854	15854	14726			7.66	7.66										
		5	16074	16074	15522			3.56	3.56										
40	**8**	1	14758	14428	13520			9.16	6.72			17.35	15.86	40.30	37.43				
		2	13942	13708	11036			26.33	24.21										
		3	13592	13314	9688			40.30	37.43										
		4	14094	14094	13172			7.00	7.00										
		5	14432	14432	13884			3.95	3.95										

Table E.9 (continued): Computational results for *No Wait Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
55	**2**	1	64880	64880	64880			0.00	0.00			0.08	0.08	0.42	0.42				
		2	50282	50282	50282			0.00	0.00										
		3	45608	45608	45416			0.42	0.42										
		4	60014	60014	60014			0.00	0.00										
		5	63258	63258	63258			0.00	0.00										
55	**4**	1	36312	36360	35760			1.54	1.68			4.48	4.62	9.81	9.81				
		2	29744	29816	27714			7.32	7.58										
		3	27488	27488	25032			9.81	9.81										
		4	34310	34406	33078			3.72	4.01										
		5	34866	34866	34866			0.00	0.00										
55	**6**	1	28172	28172	26720			5.43	5.43			11.88	12.60	27.28	27.72				
		2	24758	25082	21266			16.42	17.94										
		3	23806	23888	18704			27.28	27.72										
		4	26854	27356	25382			5.80	7.78										
		5	27214	27134	26052			4.46	4.15										
55	**8**	1	24622	24128	22720			8.37	6.20			18.14	16.46	39.03	36.39				
		2	23368	22992	18414			26.90	24.86										
		3	22578	22150	16240			39.03	36.39										
		4	23738	23468	21978			8.01	6.78										
		5	24014	23936	22152			8.41	8.05										

Table E.9 (continued): Computational results for *No Wait Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
10	**2**	1	1560	1560	1560	1560		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1440	1440	1440	1440		0.00	0.00	0.00	0.00								
		3	1380	1380	1380	1380		0.00	0.00	0.00	0.00								
		4	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
		5	1560	1560	1560	1560		0.00	0.00	0.00	0.00								
10	**4**	1	994	994	936	960	2796.81	6.20	6.20	3.54	3.54	5.93	5.93	10.83	10.83	1.65	1.65	4.72	4.72
		2	1064	1064	960	1016	9737.77	10.83	10.83	4.72	4.72								
		3	880	880	828	880	2819.89	6.28	6.28	0.00	0.00								
		4	1054	1054	1040	1054	3000.27	1.35	1.35	0.00	0.00								
		5	1092	1092	1040	1092	6470.92	5.00	5.00	0.00	0.00								
10	**6**	1	994	994	832	960	2796.81	19.47	19.47	3.54	3.54	17.06	17.06	23.15	23.15	1.65	1.65	4.72	4.72
		2	1064	1064	864	1016	6583.86	23.15	23.15	4.72	4.72								
		3	880	880	736	880	2235.39	19.57	19.57	0.00	0.00								
		4	1054	1054	936	1054	3000.27	12.61	12.61	0.00	0.00								
		5	1092	1092	988	1092	6470.92	10.53	10.53	0.00	0.00								
10	**8**	1	994	994	832	960	2796.81	19.47	19.47	3.54	3.54	17.06	17.06	23.15	23.15	1.65	1.65	4.72	4.72
		2	1064	1064	864	1016	11335.24	23.15	23.15	4.72	4.72								
		3	880	880	736	880	2357.05	19.57	19.57	0.00	0.00								
		4	1054	1054	936	1054	3000.27	12.61	12.61	0.00	0.00								
		5	1092	1092	988	1092	6470.92	10.53	10.53	0.00	0.00								

Table E.10: Computational results for *No Wait Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
25	**2**	1	8788	8788	8788			0.00	0.00			0.00	0.00	0.00	0.00				
		2	8112	8112	8112			0.00	0.00										
		3	7774	7774	7774			0.00	0.00										
		4	8788	8788	8788			0.00	0.00										
		5	8788	8788	8788			0.00	0.00										
25	**4**	1	4760	4760	4732			0.59	0.59			0.85	0.57	2.23	1.29				
		2	4760	4716	4656			2.23	1.29										
		3	4214	4204	4186			0.67	0.43										
		4	4768	4758	4732			0.76	0.55										
		5	4732	4732	4732			0.00	0.00										
25	**6**	1	3606	3500	3380			6.69	3.55			7.62	6.13	12.99	11.69				
		2	4176	4128	3696			12.99	11.69										
		3	3256	3208	2990			8.90	7.29										
		4	3970	3916	3796			4.58	3.16										
		5	3984	3984	3796			4.95	4.95										
25	**8**	1	3606	3500	3016			19.56	16.05			21.13	19.19	29.85	28.36				
		2	4176	4128	3216			29.85	28.36										
		3	3256	3208	2668			22.04	20.24										
		4	3810	3714	3328			14.48	11.60										
		5	3984	3984	3328			19.71	19.71										

Table E.10 (continued): Computational results for *No Wait Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
40	**2**	1	21840	21840	21840			0.00	0.00			0.00	0.00	0.00	0.00				
		2	20160	20160	20160			0.00	0.00										
		3	19320	19320	19320			0.00	0.00										
		4	21840	21840	21840			0.00	0.00										
		5	21840	21840	21840			0.00	0.00										
40	**4**	1	11488	11488	11440			0.42	0.42			0.57	0.45	1.14	0.71				
		2	10680	10632	10560			1.14	0.68										
		3	10200	10192	10120			0.79	0.71										
		4	11496	11488	11440			0.49	0.42										
		5	11440	11440	11440			0.00	0.00										
40	**6**	1	8116	8116	8008			1.35	1.35			3.90	2.62	7.94	4.63				
		2	8290	7942	7680			7.94	3.41										
		3	7458	7412	7084			5.28	4.63										
		4	8804	8752	8632			1.99	1.39										
		5	8564	8512	8320			2.93	2.31										
40	**8**	1	7040	6986	6240			12.82	11.96			15.48	14.70	23.76	23.76				
		2	7642	7594	6480			17.93	17.19										
		3	7116	7116	5750			23.76	23.76										
		4	7842	7786	7280			7.72	6.95										
		5	8084	7978	7020			15.16	13.65										

Table E.10 (continued): Computational results for *No Wait Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
55	**2**	1	40768	40768	40768			0.00	0.00			0.00	0.00	0.00	0.00				
		2	37632	37632	37632			0.00	0.00										
		3	36064	36064	36064			0.00	0.00										
		4	40768	40768	40768			0.00	0.00										
		5	40768	40768	40768			0.00	0.00										
55	**4**	1	21216	21202	21112			0.49	0.43			0.44	0.41	0.85	0.78				
		2	19654	19640	19488			0.85	0.78										
		3	18758	18758	18676			0.44	0.44										
		4	21202	21202	21112			0.43	0.43										
		5	21112	21112	21112			0.00	0.00										
55	**6**	1	15094	14736	14560			3.67	1.21			2.62	2.54	3.95	4.08				
		2	14420	14372	13872			3.95	3.60										
		3	13324	13406	12880			3.45	4.08										
		4	15264	15248	15028			1.57	1.46										
		5	14624	14902	14560			0.44	2.35										
55	**8**	1	12490	12276	11284			10.69	8.79			10.38	8.95	15.05	13.19				
		2	13088	12876	11376			15.05	13.19										
		3	11282	11158	9982			13.02	11.78										
		4	13212	12890	12324			7.21	4.59										
		5	12282	12336	11596			5.92	6.38										

Table E.10 (continued): Computational results for *No Wait Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
10	**2**	1	2160	2160	2160	2160		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	2040	2040	2040	2040		0.00	0.00	0.00	0.00								
		3	1980	1980	1980	1980		0.00	0.00	0.00	0.00								
		4	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
		5	2160	2160	2160	2160		0.00	0.00	0.00	0.00								
10	**4**	1	1296	1296	1296	1296		0.00	0.00	0.00	0.00	0.58	0.58	1.01	1.01	0.00	0.00	0.00	0.00
		2	1236	1236	1224	1236	5167.71	0.98	0.98	0.00	0.00								
		3	1200	1200	1188	1200	2626.64	1.01	1.01	0.00	0.00								
		4	1308	1308	1296	1308	1499.70	0.93	0.93	0.00	0.00								
		5	1440	1440	1440	1440		0.00	0.00	0.00	0.00								
10	**6**	1	1296	1236	1152	1236	1316.86	12.50	7.29	4.85	0.00	11.77	10.73	14.34	14.34	3.06	2.09	5.26	5.26
		2	1236	1236	1088	1236	5167.71	13.60	13.60	0.00	0.00								
		3	1132	1132	990	1076	1270.04	14.34	14.34	5.20	5.20								
		4	1236	1236	1152	1236	926.56	7.29	7.29	0.00	0.00								
		5	1440	1440	1296	1368	2934.76	11.11	11.11	5.26	5.26								
10	**8**	1	1296	1236	1152	1236	1316.86	12.50	7.29	4.85	0.00	13.08	12.04	17.65	17.65	3.06	2.09	5.26	5.26
		2	1236	1236	1088	1236	5167.71	13.60	13.60	0.00	0.00								
		3	1132	1132	990	1076	1270.04	14.34	14.34	5.20	5.20								
		4	1236	1236	1152	1236	926.56	7.29	7.29	0.00	0.00								
		5	1440	1440	1224	1368	2934.76	17.65	17.65	5.26	5.26								

Table E.11: Computational results for *No Wait Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
25	**2**	1	12168	12168	12168			0.00	0.00			0.00	0.00	0.00	0.00				
		2	11492	11492	11492			0.00	0.00										
		3	11154	11154	11154			0.00	0.00										
		4	12168	12168	12168			0.00	0.00										
		5	12168	12168	12168			0.00	0.00										
25	**4**	1	6562	6562	6552			0.15	0.15			0.21	0.21	0.58	0.58				
		2	6224	6224	6188			0.58	0.58										
		3	6016	6016	6006			0.17	0.17										
		4	6562	6562	6552			0.15	0.15										
		5	6552	6552	6552			0.00	0.00										
25	**6**	1	4720	4706	4680			0.85	0.56			1.60	1.19	2.58	2.58				
		2	5092	5092	4964			2.58	2.58										
		3	4386	4386	4290			2.24	2.24										
		4	5012	4996	4968			0.89	0.56										
		5	5040	4968	4968			1.45	0.00										
25	**8**	1	4306	4306	3960			8.74	8.74			8.63	7.99	14.66	13.10				
		2	4990	4922	4352			14.66	13.10										
		3	3974	3974	3630			9.48	9.48										
		4	4456	4456	4392			1.46	1.46										
		5	4780	4708	4392			8.83	7.19										

Table E.11 (continued): Computational results for *No Wait Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
40	**2**	1	30240	30240	30240			0.00	0.00			0.00	0.00	0.00	0.00				
		2	28560	28560	28560			0.00	0.00										
		3	27720	27720	27720			0.00	0.00										
		4	30240	30240	30240			0.00	0.00										
		5	30240	30240	30240			0.00	0.00										
40	**4**	1	15848	15848	15840			0.05	0.05			0.09	0.09	0.21	0.21				
		2	14992	14992	14960			0.21	0.21										
		3	14536	14536	14520			0.11	0.11										
		4	15848	15848	15840			0.05	0.05										
		5	15840	15840	15840			0.00	0.00										
40	**6**	1	11140	11140	11088			0.47	0.47			0.73	0.52	1.60	0.92				
		2	10640	10568	10472			1.60	0.92										
		3	10248	10228	10164			0.83	0.63										
		4	11152	11152	11088			0.58	0.58										
		5	11108	11088	11088			0.18	0.00										
40	**8**	1	8864	8720	8640			2.59	0.93			3.64	3.15	7.60	7.60				
		2	9312	9312	8840			5.34	5.34										
		3	8522	8522	7920			7.60	7.60										
		4	9484	9484	9360			1.32	1.32										
		5	9484	9412	9360			1.32	0.56										

Table E.11 (continued): Computational results for *No Wait Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
55	**2**	1	56448	56448	56448			0.00	0.00			0.00	0.00	0.00	0.00				
		2	53312	53312	53312			0.00	0.00										
		3	51744	51744	51744			0.00	0.00										
		4	56448	56448	56448			0.00	0.00										
		5	56448	56448	56448			0.00	0.00										
55	**4**	1	29254	29254	29232			0.08	0.08			0.07	0.07	0.16	0.16				
		2	27652	27652	27608			0.16	0.16										
		3	26810	26810	26796			0.05	0.05										
		4	29246	29246	29232			0.05	0.05										
		5	29232	29232	29232			0.00	0.00										
55	**6**	1	20286	20274	20160			0.63	0.57			0.58	0.47	1.22	0.79				
		2	19272	19190	19040			1.22	0.79										
		3	18570	18556	18480			0.49	0.41										
		4	20274	20274	20160			0.57	0.57										
		5	20160	20160	20160			0.00	0.00										
55	**8**	1	15808	15996	15624			1.18	2.38			1.69	1.77	3.31	2.44				
		2	16158	16022	15640			3.31	2.44										
		3	14662	14596	14322			2.37	1.91										
		4	16744	16726	16560			1.11	1.00										
		5	15700	15800	15624			0.49	1.13										

Table E.11 (continued): Computational results for *No Wait Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
10	**2**	1	2400	2400	2400	2400		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860	1860		0.00	0.00	0.00	0.00								
		3	1680	1680	1680	1680		0.00	0.00	0.00	0.00								
		4	2220	2220	2220	2220		0.00	0.00	0.00	0.00								
		5	2340	2340	2340	2340		0.00	0.00	0.00	0.00								
10	**4**	1	1440	1440	1440	1440		0.00	0.00	0.00	0.00	2.71	2.71	8.78	8.78	0.00	0.00	0.00	0.00
		2	1214	1214	1116	1214	5615.35	8.78	8.78	0.00	0.00								
		3	1056	1056	1008	1056	1776.71	4.76	4.76	0.00	0.00								
		4	1332	1332	1332	1332		0.00	0.00	0.00	0.00								
		5	1404	1404	1404	1404		0.00	0.00	0.00	0.00								
10	**6**	1	1300	1300	1200	1240	709.58	8.33	8.33	4.84	4.84	17.03	17.03	31.63	31.63	1.23	1.23	4.84	4.84
		2	1214	1214	992	1214	6216.62	22.38	22.38	0.00	0.00								
		3	1032	1032	784	1032	1707.36	31.63	31.63	0.00	0.00								
		4	1306	1306	1184	1306	1246.28	10.30	10.30	0.00	0.00								
		5	1404	1404	1248	1386	3096.17	12.50	12.50	1.30	1.30								
10	**8**	1	1300	1300	1200	1240	709.58	8.33	8.33	4.84	4.84	17.03	17.03	31.63	31.63	1.23	1.23	4.84	4.84
		2	1214	1214	992	1214	2875.94	22.38	22.38	0.00	0.00								
		3	1032	1032	784	1032	863.08	31.63	31.63	0.00	0.00								
		4	1306	1306	1184	1306	1246.28	10.30	10.30	0.00	0.00								
		5	1404	1404	1248	1386	3096.17	12.50	12.50	1.30	1.30								

Table E.12: Computational results for *No Wait Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
25	**2**	1	13520	13520	13520			0.00	0.00			0.00	0.00	0.00	0.00				
		2	10478	10478	10478			0.00	0.00										
		3	9464	9464	9464			0.00	0.00										
		4	12506	12506	12506			0.00	0.00										
		5	13182	13182	13182			0.00	0.00										
25	**4**	1	7280	7280	7280			0.00	0.00			0.71	0.62	2.76	2.34				
		2	5798	5774	5642			2.76	2.34										
		3	5136	5136	5096			0.78	0.78										
		4	6734	6734	6734			0.00	0.00										
		5	7098	7098	7098			0.00	0.00										
25	**6**	1	5252	5252	5200			1.00	1.00			8.04	7.61	20.15	18.00				
		2	5140	5048	4278			20.15	18.00										
		3	3960	3960	3640			8.79	8.79										
		4	5066	5066	4810			5.32	5.32										
		5	5320	5320	5070			4.93	4.93										
25	**8**	1	4700	4700	4160			12.98	12.98			19.37	18.34	33.10	31.62				
		2	4978	4978	3782			31.62	31.62										
		3	3876	3776	2912			33.10	29.67										
		4	4728	4654	4292			10.16	8.43										
		5	4930	4930	4524			8.97	8.97										

Table E.12 (continued): Computational results for *No Wait Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
40	**2**	1	33600	33600	33600			0.00	0.00			0.00	0.00	0.00	0.00				
		2	26040	26040	26040			0.00	0.00										
		3	23520	23520	23520			0.00	0.00										
		4	31080	31080	31080			0.00	0.00										
		5	32760	32760	32760			0.00	0.00										
40	**4**	1	17600	17600	17600			0.00	0.00			0.51	0.51	1.30	1.30				
		2	13784	13784	13640			1.06	1.06										
		3	12480	12480	12320			1.30	1.30										
		4	16312	16312	16280			0.20	0.20										
		5	17160	17160	17160			0.00	0.00										
40	**6**	1	12400	12384	12320			0.65	0.52			3.46	3.43	8.49	8.49				
		2	10112	10112	9548			5.91	5.91										
		3	9356	9356	8624			8.49	8.49										
		4	11652	11652	11396			2.25	2.25										
		5	12012	12012	12012			0.00	0.00										
40	**8**	1	9792	9968	9600			2.00	3.83			11.27	10.87	25.89	24.94				
		2	9356	9134	7750			20.72	17.86										
		3	8460	8396	6720			25.89	24.94										
		4	10116	10116	9620			5.16	5.16										
		5	10000	10000	9750			2.56	2.56										

Table E.12 (continued): Computational results for *No Wait Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

n	K	Ins. #	NWH1	NWH2	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)		Optimality Gap (%)		Average LB Gap (%)		Maximum LB Gap (%)		Average Optimality Gap (%)		Maximum Optimality Gap (%)	
								NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2	NWH1	NWH2
55	**2**	1	62720	62720	62720			0.00	0.00			0.00	0.00	0.00	0.00				
		2	48608	48608	48608			0.00	0.00										
		3	43904	43904	43904			0.00	0.00										
		4	58016	58016	58016			0.00	0.00										
		5	61152	61152	61152			0.00	0.00										
55	**4**	1	32480	32480	32480			0.00	0.00			0.33	0.31	0.79	0.77				
		2	25370	25352	25172			0.79	0.72										
		3	22912	22912	22736			0.77	0.77										
		4	30068	30068	30044			0.08	0.08										
		5	31668	31668	31668			0.00	0.00										
55	**6**	1	22580	22552	22400			0.80	0.68			2.23	2.02	4.56	4.22				
		2	18152	18092	17360			4.56	4.22										
		3	16240	16192	15680			3.57	3.27										
		4	21176	21120	20720			2.20	1.93										
		5	21840	21840	21840			0.00	0.00										
55	**8**	1	17698	17698	17360			1.95	1.95			6.99	6.24	13.41	12.51				
		2	15680	15556	13826			13.41	12.51										
		3	13774	13424	12152			13.35	10.47										
		4	17390	17390	16502			5.38	5.38										
		5	17076	17076	16926			0.89	0.89										

Table E.12 (continued): Computational results for *No Wait Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

Appendix F

Numerical Results for the General Problem

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
10	**2**	1	1560	1560	1560		0.00	0.00	0.00	0.00	0.00	0.00
		2	1440	1440	1440		0.00	0.00				
		3	1380	1380	1380		0.00	0.00				
		4	1560	1560	1560		0.00	0.00				
		5	1560	1560	1560		0.00	0.00				
10	**4**	1	936	936	936		0.00	0.00	0.00	0.00	0.00	0.00
		2	864	864	864		0.00	0.00				
		3	828	828	828		0.00	0.00				
		4	936	936	936		0.00	0.00				
		5	936	936	936		0.00	0.00				
10	**6**	1	728	728	728		0.00	0.00	0.55	2.75	0.00	0.00
		2	672	672	672		0.00	0.00				
		3	644	644	644		0.00	0.00				
		4	728	728	728		0.00	0.00				
		5	748	728	748	443.07	2.75	0.00				
10	**8**	1	686	624	686	124.91	9.94	0.00	14.27	19.87	0.00	0.00
		2	672	576	672	460.94	16.67	0.00				
		3	608	552	608	248.54	10.14	0.00				
		4	716	624	716	271.69	14.74	0.00				
		5	748	624	748	443.07	19.87	0.00				

Table F.1: Computational results for *General Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
25	**2**	1	8788	8788			0.00		0.00	0.00		
		2	8112	8112			0.00					
		3	7774	7774			0.00					
		4	8788	8788			0.00					
		5	8788	8788			0.00					
25	**4**	1	4732	4732			0.00		0.00	0.00		
		2	4368	4368			0.00					
		3	4186	4186			0.00					
		4	4732	4732			0.00					
		5	4732	4732			0.00					
25	**6**	1	3380	3380			0.00		0.09	0.45		
		2	3134	3120			0.45					
		3	2990	2990			0.00					
		4	3380	3380			0.00					
		5	3380	3380			0.00					
25	**8**	1	2722	2704			0.67		1.49	5.37		
		2	2630	2496			5.37					
		3	2410	2392			0.75					
		4	2722	2704			0.67					
		5	2704	2704			0.00					

Table F.1 (continued): Computational results for *General Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
40	**2**	1	21840	21840			0.00		0.00	0.00		
		2	20160	20160			0.00					
		3	19320	19320			0.00					
		4	21840	21840			0.00					
		5	21840	21840			0.00					
40	**4**	1	11440	11440			0.00		0.00	0.00		
		2	10560	10560			0.00					
		3	10120	10120			0.00					
		4	11440	11440			0.00					
		5	11440	11440			0.00					
40	**6**	1	8008	8008			0.00		0.00	0.00		
		2	7392	7392			0.00					
		3	7084	7084			0.00					
		4	8008	8008			0.00					
		5	8008	8008			0.00					
40	**8**	1	6256	6240			0.26		0.33	0.58		
		2	5792	5760			0.56					
		3	5552	5520			0.58					
		4	6256	6240			0.26					
		5	6240	6240			0.00					

Table F.1 (continued): Computational results for *General Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
55	**2**	1	40768	40768			0.00		0.00	0.00		
		2	37632	37632			0.00					
		3	36064	36064			0.00					
		4	40768	40768			0.00					
		5	40768	40768			0.00					
55	**4**	1	21112	21112			0.00		0.00	0.00		
		2	19488	19488			0.00					
		3	18676	18676			0.00					
		4	21112	21112			0.00					
		5	21112	21112			0.00					
55	**6**	1	14560	14560			0.00		0.00	0.00		
		2	13440	13440			0.00					
		3	12880	12880			0.00					
		4	14560	14560			0.00					
		5	14560	14560			0.00					
55	**8**	1	11298	11284			0.12		0.14	0.29		
		2	10446	10416			0.29					
		3	9996	9982			0.14					
		4	11298	11284			0.12					
		5	11284	11284			0.00					

Table F.1 (continued): Computational results for *General Problem* where $p_j \in [1,11]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
10	**2**	1	2160	2160	2160		0.00	0.00	0.00	0.00	0.00	0.00
		2	2040	2040	2040		0.00	0.00				
		3	1980	1980	1980		0.00	0.00				
		4	2160	2160	2160		0.00	0.00				
		5	2160	2160	2160		0.00	0.00				
10	**4**	1	1296	1296	1296		0.00	0.00	0.00	0.00	0.00	0.00
		2	1224	1224	1224		0.00	0.00				
		3	1188	1188	1188		0.00	0.00				
		4	1296	1296	1296		0.00	0.00				
		5	1296	1296	1296		0.00	0.00				
10	**6**	1	1008	1008	1008		0.00	0.00	0.00	0.00	0.00	0.00
		2	952	952	952		0.00	0.00				
		3	924	924	924		0.00	0.00				
		4	1008	1008	1008		0.00	0.00				
		5	1008	1008	1008		0.00	0.00				
10	**8**	1	864	864	864		0.00	0.00	4.30	8.33	0.00	0.00
		2	884	816	884	223.77	8.33	0.00				
		3	812	792	812	256.36	2.53	0.00				
		4	884	864	884	464.63	2.31	0.00				
		5	936	864	936	127.09	8.33	0.00				

Table F.2: Computational results for *General Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
25	**2**	1	12168	12168			0.00		0.00	0.00		
		2	11492	11492			0.00					
		3	11154	11154			0.00					
		4	12168	12168			0.00					
		5	12168	12168			0.00					
25	**4**	1	6552	6552			0.00		0.00	0.00		
		2	6188	6188			0.00					
		3	6006	6006			0.00					
		4	6552	6552			0.00					
		5	6552	6552			0.00					
25	**6**	1	4680	4680			0.00		0.00	0.00		
		2	4420	4420			0.00					
		3	4290	4290			0.00					
		4	4680	4680			0.00					
		5	4680	4680			0.00					
25	**8**	1	3744	3744			0.00		0.05	0.25		
		2	3545	3536			0.25					
		3	3432	3432			0.00					
		4	3744	3744			0.00					
		5	3744	3744			0.00					

Table F.2 (continued): Computational results for *General Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
40	**2**	1	30240	30240			0.00		0.00	0.00		
		2	28560	28560			0.00					
		3	27720	27720			0.00					
		4	30240	30240			0.00					
		5	30240	30240			0.00					
40	**4**	1	15840	15840			0.00		0.00	0.00		
		2	14960	14960			0.00					
		3	14520	14520			0.00					
		4	15840	15840			0.00					
		5	15840	15840			0.00					
40	**6**	1	11088	11088			0.00		0.00	0.00		
		2	10472	10472			0.00					
		3	10164	10164			0.00					
		4	11088	11088			0.00					
		5	11088	11088			0.00					
40	**8**	1	8640	8640			0.00		0.00	0.00		
		2	8160	8160			0.00					
		3	7920	7920			0.00					
		4	8640	8640			0.00					
		5	8640	8640			0.00					

Table F.2 (continued): Computational results for *General Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
55	**2**	1	56448	56448			0.00		0.00	0.00		
		2	53312	53312			0.00					
		3	51744	51744			0.00					
		4	56448	56448			0.00					
		5	56448	56448			0.00					
55	**4**	1	29232	29232			0.00		0.00	0.00		
		2	27608	27608			0.00					
		3	26796	26796			0.00					
		4	29232	29232			0.00					
		5	29232	29232			0.00					
55	**6**	1	20160	20160			0.00		0.00	0.00		
		2	19040	19040			0.00					
		3	18480	18480			0.00					
		4	20160	20160			0.00					
		5	20160	20160			0.00					
55	**8**	1	15624	15624			0.00		0.00	0.00		
		2	14756	14756			0.00					
		3	14322	14322			0.00					
		4	15624	15624			0.00					
		5	15624	15624			0.00					

Table F.2 (continued): Computational results for *General Problem* where $p_j \in [1,11]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
10	**2**	1	2400	2400	2400		0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860		0.00	0.00				
		3	1680	1680	1680		0.00	0.00				
		4	2220	2220	2220		0.00	0.00				
		5	2340	2340	2340		0.00	0.00				
10	**4**	1	1440	1440	1440		0.00	0.00	0.00	0.00	0.00	0.00
		2	1116	1116	1116		0.00	0.00				
		3	1008	1008	1008		0.00	0.00				
		4	1332	1332	1332		0.00	0.00				
		5	1404	1404	1404		0.00	0.00				
10	**6**	1	1120	1120	1120		0.00	0.00	0.00	0.00	0.00	0.00
		2	868	868	868		0.00	0.00				
		3	784	784	784		0.00	0.00				
		4	1036	1036	1036		0.00	0.00				
		5	1092	1092	1092		0.00	0.00				
10	**8**	1	960	960	960		0.00	0.00	5.65	12.37	0.00	0.00
		2	836	744	836	132.34	12.37	0.00				
		3	728	672	728	163.90	8.33	0.00				
		4	898	888	898	265.87	1.13	0.00				
		5	996	936	996	160.56	6.41	0.00				

Table F.3: Computational results for *General Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
25	**2**	1	13520	13520			0.00		0.00	0.00		
		2	10478	10478			0.00					
		3	9464	9464			0.00					
		4	12506	12506			0.00					
		5	13182	13182			0.00					
25	**4**	1	7280	7280			0.00		0.00	0.00		
		2	5642	5642			0.00					
		3	5096	5096			0.00					
		4	6734	6734			0.00					
		5	7098	7098			0.00					
25	**6**	1	5200	5200			0.00		0.00	0.00		
		2	4030	4030			0.00					
		3	3640	3640			0.00					
		4	4810	4810			0.00					
		5	5070	5070			0.00					
25	**8**	1	4160	4160			0.00		0.33	1.67		
		2	3278	3224			1.67					
		3	2912	2912			0.00					
		4	3848	3848			0.00					
		5	4056	4056			0.00					

Table F.3 (continued): Computational results for *General Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
40	**2**	1	33600	33600			0.00		0.00	0.00		
		2	26040	26040			0.00					
		3	23520	23520			0.00					
		4	31080	31080			0.00					
		5	32760	32760			0.00					
40	**4**	1	17600	17600			0.00		0.00	0.00		
		2	13640	13640			0.00					
		3	12320	12320			0.00					
		4	16280	16280			0.00					
		5	17160	17160			0.00					
40	**6**	1	12320	12320			0.00		0.00	0.00		
		2	9548	9548			0.00					
		3	8624	8624			0.00					
		4	11396	11396			0.00					
		5	12012	12012			0.00					
40	**8**	1	9600	9600			0.00		0.02	0.12		
		2	7440	7440			0.00					
		3	6728	6720			0.12					
		4	8880	8880			0.00					
		5	9360	9360			0.00					

Table F.3 (continued): Computational results for *General Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
55	**2**	1	62720	62720			0.00		0.00	0.00		
		2	48608	48608			0.00					
		3	43904	43904			0.00					
		4	58016	58016			0.00					
		5	61152	61152			0.00					
55	**4**	1	32480	32480			0.00		0.00	0.00		
		2	25172	25172			0.00					
		3	22736	22736			0.00					
		4	30044	30044			0.00					
		5	31668	31668			0.00					
55	**6**	1	22400	22400			0.00		0.00	0.00		
		2	17360	17360			0.00					
		3	15680	15680			0.00					
		4	20720	20720			0.00					
		5	21840	21840			0.00					
55	**8**	1	17360	17360			0.00		0.00	0.00		
		2	13454	13454			0.00					
		3	12152	12152			0.00					
		4	16058	16058			0.00					
		5	16926	16926			0.00					

Table F.3 (continued): Computational results for *General Problem* where $p_j \in [1,11]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
10	**2**	1	1560	1560	1560		0.00	0.00	0.00	0.00	0.00	0.00
		2	1440	1440	1440		0.00	0.00				
		3	1380	1380	1380		0.00	0.00				
		4	1560	1560	1560		0.00	0.00				
		5	1560	1560	1560		0.00	0.00				
10	**4**	1	936	936	936		0.00	0.00	2.14	4.17	0.00	0.00
		2	900	864	900	3667.16	4.17	0.00				
		3	838	828	838	1989.84	1.21	0.00				
		4	956	936	956	1977.28	2.14	0.00				
		5	966	936	966	3129.95	3.21	0.00				
10	**6**	1	844	728	844	813.64	15.93	0.00	19.05	22.92	0.10	0.52
		2	826	672	826	1161.52	22.92	0.00				
		3	770	644	766	637.23	19.57	0.52				
		4	862	728	862	615.01	18.41	0.00				
		5	862	728	862	965.10	18.41	0.00				
10	**8**	1	844	624	844	813.64	35.26	0.00	38.89	43.40	0.10	0.52
		2	826	576	826	952.30	43.40	0.00				
		3	770	552	766	1929.22	39.49	0.52				
		4	862	624	862	615.01	38.14	0.00				
		5	862	624	862	965.10	38.14	0.00				

Table F.4: Computational results for *General Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
25	**2**	1	8788	8788			0.00		0.00	0.00		
		2	8112	8112			0.00					
		3	7774	7774			0.00					
		4	8788	8788			0.00					
		5	8788	8788			0.00					
25	**4**	1	4732	4732			0.00		0.34	1.72		
		2	4443	4368			1.72					
		3	4186	4186			0.00					
		4	4732	4732			0.00					
		5	4732	4732			0.00					
25	**6**	1	3666	3380			8.46		11.59	16.06		
		2	3621	3120			16.06					
		3	3315	2990			10.87					
		4	3770	3380			11.54					
		5	3752	3380			11.01					
25	**8**	1	3262	2704			20.64		23.90	30.05		
		2	3246	2496			30.05					
		3	2954	2392			23.49					
		4	3337	2704			23.41					
		5	3297	2704			21.93					

Table F.4 (continued): Computational results for *General Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
40	**2**	1	21840	21840			0.00		0.00	0.00		
		2	20160	20160			0.00					
		3	19320	19320			0.00					
		4	21840	21840			0.00					
		5	21840	21840			0.00					
40	**4**	1	11440	11440			0.00		0.00	0.00		
		2	10560	10560			0.00					
		3	10120	10120			0.00					
		4	11440	11440			0.00					
		5	11440	11440			0.00					
40	**6**	1	8308	8008			3.75		6.29	7.98		
		2	7914	7392			7.06					
		3	7600	7084			7.28					
		4	8647	8008			7.98					
		5	8440	8008			5.39					
40	**8**	1	7112	6240			13.97		15.88	17.21		
		2	6748	5760			17.15					
		3	6438	5520			16.63					
		4	7314	6240			17.21					
		5	7140	6240			14.42					

Table F.4 (continued): Computational results for *General Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
55	**2**	1	40768	40768			0.00		0.00	0.00		
		2	37632	37632			0.00					
		3	36064	36064			0.00					
		4	40768	40768			0.00					
		5	40768	40768			0.00					
55	**4**	1	21112	21112			0.00		0.00	0.00		
		2	19488	19488			0.00					
		3	18676	18676			0.00					
		4	21112	21112			0.00					
		5	21112	21112			0.00					
55	**6**	1	14983	14560			2.91		4.48	5.68		
		2	14204	13440			5.68					
		3	13484	12880			4.69					
		4	15366	14560			5.54					
		5	15083	14560			3.59					
55	**8**	1	12460	11284			10.42		11.85	13.32		
		2	11800	10416			13.29					
		3	11187	9982			12.07					
		4	12787	11284			13.32					
		5	12429	11284			10.15					

Table F.4 (continued): Computational results for *General Problem* where $p_j \in [5,15]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
10	**2**	1	2160	2160	2160		0.00	0.00	0.00	0.00	0.00	0.00
		2	2040	2040	2040		0.00	0.00				
		3	1980	1980	1980		0.00	0.00				
		4	2160	2160	2160		0.00	0.00				
		5	2160	2160	2160		0.00	0.00				
10	**4**	1	1296	1296	1296		0.00	0.00	0.00	0.00	0.00	0.00
		2	1224	1224	1224		0.00	0.00				
		3	1188	1188	1188		0.00	0.00				
		4	1296	1296	1296		0.00	0.00				
		5	1296	1296	1296		0.00	0.00				
10	**6**	1	1078	1008	1078	960.90	6.94	0.00	8.70	11.11	0.00	0.00
		2	1055	952	1055	717.96	10.82	0.00				
		3	984	924	984	731.85	6.49	0.00				
		4	1090	1008	1090	700.67	8.13	0.00				
		5	1120	1008	1120	675.74	11.11	0.00				
10	**8**	1	1078	864	1078	1402.63	24.77	0.00	26.82	29.63	0.00	0.00
		2	1055	816	1055	519.41	29.29	0.00				
		3	984	792	984	312.95	24.24	0.00				
		4	1090	864	1090	700.67	26.16	0.00				
		5	1120	864	1120	675.74	29.63	0.00				

Table F.5: Computational results for *General Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
25	**2**	1	12168	12168			0.00		0.00	0.00		
		2	11492	11492			0.00					
		3	11154	11154			0.00					
		4	12168	12168			0.00					
		5	12168	12168			0.00					
25	**4**	1	6552	6552			0.00		0.00	0.00		
		2	6188	6188			0.00					
		3	6006	6006			0.00					
		4	6552	6552			0.00					
		5	6552	6552			0.00					
25	**6**	1	4680	4680			0.00		1.95	7.08		
		2	4733	4420			7.08					
		3	4290	4290			0.00					
		4	4730	4680			1.07					
		5	4755	4680			1.60					
25	**8**	1	4085	3744			9.11		13.37	19.15		
		2	4213	3536			19.15					
		3	3830	3432			11.60					
		4	4248	3744			13.46					
		5	4251	3744			13.54					

Table F.5 (continued): Computational results for *General Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
40	**2**	1	30240	30240			0.00		0.00	0.00		
		2	28560	28560			0.00					
		3	27720	27720			0.00					
		4	30240	30240			0.00					
		5	30240	30240			0.00					
40	**4**	1	15840	15840			0.00		0.00	0.00		
		2	14960	14960			0.00					
		3	14520	14520			0.00					
		4	15840	15840			0.00					
		5	15840	15840			0.00					
40	**6**	1	11088	11088			0.00		0.22	0.72		
		2	10512	10472			0.38					
		3	10164	10164			0.00					
		4	11168	11088			0.72					
		5	11088	11088			0.00					
40	**8**	1	9114	8640			5.49		8.02	9.68		
		2	8932	8160			9.46					
		3	8524	7920			7.63					
		4	9476	8640			9.68					
		5	9320	8640			7.87					

Table F.5 (continued): Computational results for *General Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
55	**2**	1	56448	56448			0.00		0.00	0.00		
		2	53312	53312			0.00					
		3	51744	51744			0.00					
		4	56448	56448			0.00					
		5	56448	56448			0.00					
55	**4**	1	29232	29232			0.00		0.00	0.00		
		2	27608	27608			0.00					
		3	26796	26796			0.00					
		4	29232	29232			0.00					
		5	29232	29232			0.00					
55	**6**	1	20160	20160			0.00		0.06	0.29		
		2	19095	19040			0.29					
		3	18480	18480			0.00					
		4	20160	20160			0.00					
		5	20160	20160			0.00					
55	**8**	1	16214	15624			3.78		5.10	6.93		
		2	15779	14756			6.93					
		3	14986	14322			4.64					
		4	16610	15624			6.31					
		5	16221	15624			3.82					

Table F.5 (continued): Computational results for *General Problem* where $p_j \in [5,15]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
10	**2**	1	2400	2400	2400		0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860		0.00	0.00				
		3	1680	1680	1680		0.00	0.00				
		4	2220	2220	2220		0.00	0.00				
		5	2340	2340	2340		0.00	0.00				
10	**4**	1	1440	1440	1440		0.00	0.00	0.00	0.00	0.00	0.00
		2	1116	1116	1116		0.00	0.00				
		3	1008	1008	1008		0.00	0.00				
		4	1332	1332	1332		0.00	0.00				
		5	1404	1404	1404		0.00	0.00				
10	**6**	1	1150	1120	1150	344.09	2.68	0.00	10.29	17.51	0.18	0.89
		2	1020	868	1020	861.57	17.51	0.00				
		3	910	784	902	664.89	16.07	0.89				
		4	1110	1036	1110	338.26	7.14	0.00				
		5	1180	1092	1180	659.10	8.06	0.00				
10	**8**	1	1150	960	1150	344.09	19.79	0.00	28.67	37.10	0.18	0.89
		2	1020	744	1020	839.97	37.10	0.00				
		3	910	672	902	852.47	35.42	0.89				
		4	1110	888	1110	338.26	25.00	0.00				
		5	1180	936	1180	659.10	26.07	0.00				

Table F.6: Computational results for *General Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
25	**2**	1	13520	13520			0.00		0.00	0.00		
		2	10478	10478			0.00					
		3	9464	9464			0.00					
		4	12506	12506			0.00					
		5	13182	13182			0.00					
25	**4**	1	7280	7280			0.00		0.00	0.00		
		2	5642	5642			0.00					
		3	5096	5096			0.00					
		4	6734	6734			0.00					
		5	7098	7098			0.00					
25	**6**	1	5200	5200			0.00		3.20	10.50		
		2	4453	4030			10.50					
		3	3821	3640			4.97					
		4	4835	4810			0.52					
		5	5070	5070			0.00					
25	**8**	1	4385	4160			5.41		15.39	24.13		
		2	4002	3224			24.13					
		3	3561	2912			22.29					
		4	4370	3848			13.57					
		5	4524	4056			11.54					

Table F.6 (continued): Computational results for *General Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
40	**2**	1	33600	33600			0.00		0.00	0.00		
		2	26040	26040			0.00					
		3	23520	23520			0.00					
		4	31080	31080			0.00					
		5	32760	32760			0.00					
40	**4**	1	17600	17600			0.00		0.00	0.00		
		2	13640	13640			0.00					
		3	12320	12320			0.00					
		4	16280	16280			0.00					
		5	17160	17160			0.00					
40	**6**	1	12320	12320			0.00		1.42	4.45		
		2	9768	9548			2.30					
		3	9008	8624			4.45					
		4	11436	11396			0.35					
		5	12012	12012			0.00					
40	**8**	1	9792	9600			2.00		9.30	16.01		
		2	8410	7440			13.04					
		3	7796	6720			16.01					
		4	9730	8880			9.57					
		5	9910	9360			5.88					

Table F.6 (continued): Computational results for *General Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
55	**2**	1	62720	62720			0.00		0.00	0.00		
		2	48608	48608			0.00					
		3	43904	43904			0.00					
		4	58016	58016			0.00					
		5	61152	61152			0.00					
55	**4**	1	32480	32480			0.00		0.00	0.00		
		2	25172	25172			0.00					
		3	22736	22736			0.00					
		4	30044	30044			0.00					
		5	31668	31668			0.00					
55	**6**	1	22400	22400			0.00		0.78	2.18		
		2	17739	17360			2.18					
		3	15949	15680			1.72					
		4	20720	20720			0.00					
		5	21840	21840			0.00					
55	**8**	1	17640	17360			1.61		6.33	11.28		
		2	14852	13454			10.39					
		3	13523	12152			11.28					
		4	16960	16058			5.62					
		5	17394	16926			2.76					

Table F.6 (continued): Computational results for *General Problem* where $p_j \in [5,15]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
10	**2**	1	1834	1560	1820	181898.53	17.56	0.77	17.51	17.82		
		2	1708	1450	1708(*)		17.79	---				
		3	1648	1400	1648(*)		17.71	---				
		4	1838	1560	1838	161391.61	17.82	0.00				
		5	1820	1560	1810(*)		16.67	---				
10	**4**	1	1434	936	1434	85488.61	53.21	0.00	58.05	65.22	1.58	4.03
		2	1444	874	1388	192790.49	65.22	4.03				
		3	1354	848	1354	179757.190	59.67	0.00				
		4	1461	936	1445	74839.4	56.09	1.11				
		5	1461	936	1444	134805.22	56.09	1.18				
10	**6**	1	1434	728	1434	85488.61	96.98	0.00	102.80	111.73	1.58	4.03
		2	1444	682	1388	192790.49	111.73	4.03				
		3	1354	664	1354	179757.190	103.92	0.00				
		4	1461	728	1445	74839.4	100.69	1.11				
		5	1461	728	1444	134805.22	100.69	1.18				
10	**8**	1	1434	624	1434	85488.61	129.81	0.00	136.24	146.42	1.58	4.03
		2	1444	586	1388	192790.49	146.42	4.03				
		3	1354	572	1354	179757.190	136.71	0.00				
		4	1461	624	1445	74839.4	134.13	1.11				
		5	1461	624	1444	134805.22	134.13	1.18				

Table F.7: Computational results for *General Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
25	**2**	1	9464	8788			7.69		7.84	8.33		
		2	8809	8137			8.26					
		3	8476	7824			8.33					
		4	9476	8788			7.83					
		5	9412	8788			7.10					
25	**4**	1	6183	4732			30.66		32.20	38.79		
		2	6097	4393			38.79					
		3	5730	4236			35.27					
		4	6106	4732			29.04					
		5	6022	4732			27.26					
25	**6**	1	5532	3380			63.67		71.38	81.59		
		2	5711	3145			81.59					
		3	5421	3040			78.32					
		4	5595	3380			65.53					
		5	5672	3380			67.81					
25	**8**	1	5532	2704			104.59		113.96	126.54		
		2	5711	2521			126.54					
		3	5421	2442			121.99					
		4	5595	2704			106.92					
		5	5672	2704			109.76					

Table F.7 (continued): Computational results for *General Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
40	**2**	1	22990	21840			5.27		5.49	6.12		
		2	21380	20200			5.84					
		3	20588	19400			6.12					
		4	23034	21840			5.47					
		5	22880	21840			4.76					
40	**4**	1	13738	11440			20.09		21.98	26.48		
		2	13195	10600			24.48					
		3	12901	10200			26.48					
		4	13754	11440			20.23					
		5	13572	11440			18.64					
40	**6**	1	11777	8008			47.07		53.72	61.61		
		2	11924	7432			60.44					
		3	11578	7164			61.61					
		4	11920	8008			48.85					
		5	12062	8008			50.62					
40	**8**	1	11777	6240			88.73		97.08	106.75		
		2	11924	5800			105.59					
		3	11578	5600			106.75					
		4	11920	6240			91.03					
		5	12062	6240			93.30					

Table F.7 (continued): Computational results for *General Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
55	**2**	1	42396	40768			3.99		4.12	4.58		
		2	39371	37687			4.47					
		3	37832	36174			4.58					
		4	42436	40768			4.09					
		5	42172	40768			3.44					
55	**4**	1	24884	21112			17.87		17.72	20.85		
		2	23381	19543			19.64					
		3	22702	18786			20.85					
		4	24469	21112			15.90					
		5	24146	21112			14.37					
55	**6**	1	20499	14560			40.79		45.35	52.77		
		2	20471	13495			51.69					
		3	19845	12990			52.77					
		4	20462	14560			40.54					
		5	20523	14560			40.95					
55	**8**	1	20499	11284			81.66		87.40	96.64		
		2	20471	10471			95.50					
		3	19845	10092			96.64					
		4	20462	11284			81.34					
		5	20523	11284			81.88					

Table F.7 (continued): Computational results for *General Problem* where $p_j \in [25,35]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
10	**2**	1	2310	2160	2310	170999.89	6.94	0.00	7.84	9.09	0.00	0.00
		2	2210	2040	2210	190760.79	8.33	0.00				
		3	2160	1980	2160	81086.12	9.09	0.00				
		4	2330	2160	2330	129037.66	7.87	0.00				
		5	2310	2160	2310	81365.27	6.94	0.00				
10	**4**	1	1750	1296	1750	41103.71	35.03	0.00	36.31	38.72	0.00	0.00
		2	1682	1224	1682	39336.05	37.42	0.00				
		3	1648	1188	1648	40989.59	38.72	0.00				
		4	1770	1296	1770	21453.83	36.57	0.00				
		5	1734	1296	1734	31669.32	33.80	0.00				
10	**6**	1	1750	1008	1750	41103.71	73.61	0.00	75.84	82.68	0.49	2.43
		2	1682	952	1682	39336.05	76.68	0.00				
		3	1688	924	1648	40989.59	82.68	2.43				
		4	1756	1008	1756	42265.24	74.21	0.00				
		5	1734	1008	1734	31669.32	72.02	0.00				
10	**8**	1	1750	864	1750	41103.71	102.55	0.00	105.15	113.13	0.49	2.43
		2	1682	816	1682	39336.05	106.13	0.00				
		3	1688	792	1648	40989.59	113.13	2.43				
		4	1756	864	1756	42265.24	103.24	0.00				
		5	1734	864	1734	31669.32	100.69	0.00				

Table F.8: Computational results for *General Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
25	**2**	1	12518	12168			2.88		3.35	3.81		
		2	11917	11492			3.70					
		3	11579	11154			3.81					
		4	12568	12168			3.29					
		5	12543	12168			3.08					
25	**4**	1	7869	6552			20.10		20.78	21.76		
		2	7519	6188			21.51					
		3	7313	6006			21.76					
		4	7910	6552			20.73					
		5	7848	6552			19.78					
25	**6**	1	6800	4680			45.30		46.82	49.80		
		2	6621	4420			49.80					
		3	6426	4290			49.79					
		4	6800	4680			45.30					
		5	6736	4680			43.93					
25	**8**	1	6436	3744			71.90		77.64	84.79		
		2	6534	3536			84.79					
		3	6292	3432			83.33					
		4	6508	3744			73.82					
		5	6527	3744			74.33					

Table F.8 (continued): Computational results for *General Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
40	**2**	1	30800	30240			1.85		2.10	2.45		
		2	29200	28560			2.24					
		3	28400	27720			2.45					
		4	30880	30240			2.12					
		5	30800	30240			1.85					
40	**4**	1	18208	15840			14.95		14.38	15.22		
		2	17168	14960			14.76					
		3	16730	14520			15.22					
		4	18114	15840			14.36					
		5	17840	15840			12.63					
40	**6**	1	14544	11088			31.17		32.52	35.66		
		2	14082	10472			34.47					
		3	13788	10164			35.66					
		4	14544	11088			31.17					
		5	14432	11088			30.16					
40	**8**	1	13311	8640			54.06		59.13	64.72		
		2	13441	8160			64.72					
		3	13040	7920			64.65					
		4	13400	8640			55.09					
		5	13577	8640			57.14					

Table F.8 (continued): Computational results for *General Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
55	**2**	1	57218	56448			1.36		1.55	1.81		
		2	54192	53312			1.65					
		3	52679	51744			1.81					
		4	57328	56448			1.56					
		5	57218	56448			1.36					
55	**4**	1	32560	29232			11.38		11.21	11.62		
		2	30796	27608			11.55					
		3	29910	26796			11.62					
		4	32560	29232			11.38					
		5	32184	29232			10.10					
55	**6**	1	25088	20160			24.44		25.51	28.24		
		2	24220	19040			27.21					
		3	23698	18480			28.24					
		4	25088	20160			24.44					
		5	24836	20160			23.19					
55	**8**	1	22580	15624			44.52		47.78	53.44		
		2	22444	14756			52.10					
		3	21976	14322			53.44					
		4	22552	15624			44.34					
		5	22575	15624			44.49					

Table F.8 (continued): Computational results for *General Problem* where $p_j \in [25,35]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
10	**2**	1	2510	2400	2510	127740.02	4.58	0.00	8.90	16.07	0.00	0.00
		2	2074	1860	2072	124644.94	11.51	0.00				
		3	1950	1680	1942	179867.56	16.07	0.00				
		4	2380	2220	2380	131910.80	7.21	0.00				
		5	2460	2340	2460	168552.26	5.13	0.00				
10	**4**	1	1902	1440	1902	25370.04	32.08	0.00	41.58	58.33	0.22	0.88
		2	1648	1116	1648	58345.15	47.67	0.00				
		3	1596	1008	1582	53465.44	58.33	0.88				
		4	1852	1332	1852	24405.94	39.04	0.00				
		5	1836	1404	1836	28640.59	30.77	0.00				
10	**6**	1	1910	1120	1882	18165.78	70.54	1.49	82.10	103.57	0.47	1.49
		2	1648	868	1648	58345.15	89.86	0.00				
		3	1596	784	1582	53465.44	103.57	0.88				
		4	1848	1036	1848	20613.73	78.38	0.00				
		5	1836	1092	1836	28640.59	68.13	0.00				
10	**8**	1	1910	960	1882	18165.78	98.96	1.49	112.45	137.50	0.47	1.49
		2	1648	744	1648	58345.15	121.51	0.00				
		3	1596	672	1582	53465.44	137.50	0.88				
		4	1848	888	1848	20613.73	108.11	0.00				
		5	1836	936	1836	28640.59	96.15	0.00				

Table F.9: Computational results for *General Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
25	**2**	1	13770	13520			1.85		3.86	6.87		
		2	11032	10478			5.29					
		3	10114	9464			6.87					
		4	12881	12506			3.00					
		5	13482	13182			2.28					
25	**4**	1	8626	7280			18.49		23.77	31.04		
		2	7156	5642			26.83					
		3	6678	5096			31.04					
		4	8305	6734			23.33					
		5	8459	7098			19.17					
25	**6**	1	7257	5200			39.56		55.15	78.41		
		2	6706	4030			66.40					
		3	6494	3640			78.41					
		4	7216	4810			50.02					
		5	7166	5070			41.34					
25	**8**	1	6862	4160			64.95		86.77	115.80		
		2	6621	3224			105.37					
		3	6284	2912			115.80					
		4	6830	3848			77.49					
		5	6904	4056			70.22					

Table F.9 (continued): Computational results for *General Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
40	**2**	1	34000	33600			1.19		2.54	5.32		
		2	26800	26040			2.92					
		3	24772	23520			5.32					
		4	31680	31080			1.93					
		5	33200	32760			1.34					
40	**4**	1	19912	17600			13.14		17.78	24.37		
		2	16288	13640			19.41					
		3	15322	12320			24.37					
		4	19268	16280			18.35					
		5	19500	17160			13.64					
40	**6**	1	15706	12320			27.48		41.89	63.31		
		2	14450	9548			51.34					
		3	14084	8624			63.31					
		4	15834	11396			38.94					
		5	15418	12012			28.35					
40	**8**	1	13958	9600			45.40		66.15	97.80		
		2	13490	7440			81.32					
		3	13292	6720			97.80					
		4	13824	8880			55.68					
		5	14094	9360			50.58					

Table F.9 (continued): Computational results for *General Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
55	**2**	1	63270	62720			0.88		1.86	3.88		
		2	49653	48608			2.15					
		3	45608	43904			3.88					
		4	58841	58016			1.42					
		5	61757	61152			0.99					
55	**4**	1	35770	32480			10.13		13.94	19.33		
		2	29100	25172			15.60					
		3	27130	22736			19.33					
		4	34406	30044			14.52					
		5	34866	31668			10.10					
55	**6**	1	27649	22400			23.43		34.40	52.35		
		2	24743	17360			42.53					
		3	23888	15680			52.35					
		4	26974	20720			30.18					
		5	26972	21840			23.50					
55	**8**	1	23499	17360			35.36		53.27	81.71		
		2	22581	13454			67.84					
		3	22081	12152			81.71					
		4	23092	16058			43.80					
		5	23298	16926			37.65					

Table F.9 (continued): Computational results for *General Problem* where $p_j \in [25,35]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
10	**2**	1	1560	1560	1560		0.00	0.00	0.00	0.00	0.00	0.00
		2	1440	1440	1440		0.00	0.00				
		3	1380	1380	1380		0.00	0.00				
		4	1560	1560	1560		0.00	0.00				
		5	1560	1560	1560		0.00	0.00				
10	**4**	1	966	936	958	1542.77	3.21	0.84	9.75	16.35	0.40	0.84
		2	977	864	974	4733.44	13.08	0.31				
		3	880	828	880	4309.56	6.28	0.00				
		4	1028	936	1028	3944.96	9.83	0.00				
		5	1089	936	1080	4871.54	16.35	0.83				
10	**6**	1	966	728	958	1542.77	32.69	0.84	41.82	49.59	1.32	2.53
		2	977	672	974	4385.29	45.39	0.31				
		3	880	644	862	2002.31	36.65	2.09				
		4	1054	728	1028	3944.96	44.78	2.53				
		5	1089	728	1080	4871.54	49.59	0.83				
10	**8**	1	966	624	958	1542.77	54.81	0.84	65.46	74.52	1.32	2.53
		2	977	576	974	3337.92	69.62	0.31				
		3	880	552	862	1341.00	59.42	2.09				
		4	1054	624	1028	3944.96	68.91	2.53				
		5	1089	624	1080	4871.54	74.52	0.83				

Table F.10: Computational results for *General Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
25	**2**	1	8788	8788			0.00		0.00	0.00		
		2	8112	8112			0.00					
		3	7774	7774			0.00					
		4	8788	8788			0.00					
		5	8788	8788			0.00					
25	**4**	1	4760	4732			0.59		1.57	6.30		
		2	4643	4368			6.30					
		3	4204	4186			0.43					
		4	4758	4732			0.55					
		5	4732	4732			0.00					
25	**6**	1	3500	3380			3.55		12.99	29.84		
		2	4051	3120			29.84					
		3	3208	2990			7.29					
		4	3732	3380			10.41					
		5	3849	3380			13.88					
25	**8**	1	3500	2704			29.44		40.45	62.30		
		2	4051	2496			62.30					
		3	3208	2392			34.11					
		4	3625	2704			34.06					
		5	3849	2704			42.34					

Table F.10 (continued): Computational results for *General Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
40	**2**	1	21840	21840			0.00		0.00	0.00		
		2	20160	20160			0.00					
		3	19320	19320			0.00					
		4	21840	21840			0.00					
		5	21840	21840			0.00					
40	**4**	1	11488	11440			0.42		0.45	0.71		
		2	10632	10560			0.68					
		3	10192	10120			0.71					
		4	11488	11440			0.42					
		5	11440	11440			0.00					
40	**6**	1	8116	8008			1.35		5.06	7.44		
		2	7942	7392			7.44					
		3	7412	7084			4.63					
		4	8526	8008			6.47					
		5	8440	8008			5.39					
40	**8**	1	6902	6240			10.61		22.75	31.77		
		2	7590	5760			31.77					
		3	6999	5520			26.79					
		4	7381	6240			18.29					
		5	7880	6240			26.28					

Table F.10 (continued): Computational results for *General Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
55	**2**	1	40768	40768			0.00		0.00	0.00		
		2	37632	37632			0.00					
		3	36064	36064			0.00					
		4	40768	40768			0.00					
		5	40768	40768			0.00					
55	**4**	1	21202	21112			0.43		0.41	0.78		
		2	19640	19488			0.78					
		3	18758	18676			0.44					
		4	21202	21112			0.43					
		5	21112	21112			0.00					
55	**6**	1	14736	14560			1.21		3.73	6.28		
		2	14284	13440			6.28					
		3	13406	12880			4.08					
		4	15248	14560			4.73					
		5	14902	14560			2.35					
55	**8**	1	12258	11284			8.63		12.49	21.95		
		2	12702	10416			21.95					
		3	11158	9982			11.78					
		4	12591	11284			11.58					
		5	12244	11284			8.51					

Table F.10 (continued): Computational results for *General Problem* where $p_j \in [1,35]$ and $d_i \in [22,28]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
10	**2**	1	2160	2160	2160		0.00	0.00	0.00	0.00	0.00	0.00
		2	2040	2040	2040		0.00	0.00				
		3	1980	1980	1980		0.00	0.00				
		4	2160	2160	2160		0.00	0.00				
		5	2160	2160	2160		0.00	0.00				
10	**4**	1	1296	1296	1296		0.00	0.00	1.05	2.31	0.00	0.00
		2	1236	1224	1236	1478.51	0.98	0.00				
		3	1200	1188	1200	1747.53	1.01	0.00				
		4	1308	1296	1308	3051.37	0.93	0.00				
		5	1326	1296	1326	2284.08	2.31	0.00				
10	**6**	1	1182	1008	1182	1280.05	17.26	0.00	23.52	30.36	0.28	1.40
		2	1236	952	1236	1478.51	29.83	0.00				
		3	1086	924	1071	554.78	17.53	1.40				
		4	1236	1008	1236	605.87	22.62	0.00				
		5	1314	1008	1314	2086.30	30.36	0.00				
10	**8**	1	1182	864	1182	1280.05	36.81	0.00	44.11	52.08	0.28	1.40
		2	1236	816	1236	1478.51	51.47	0.00				
		3	1086	792	1071	554.78	37.12	1.40				
		4	1236	864	1236	605.87	43.06	0.00				
		5	1314	864	1314	2086.30	52.08	0.00				

Table F.11: Computational results for *General Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
25	**2**	1	12168	12168			0.00		0.00	0.00		
		2	11492	11492			0.00					
		3	11154	11154			0.00					
		4	12168	12168			0.00					
		5	12168	12168			0.00					
25	**4**	1	6562	6552			0.15		0.21	0.58		
		2	6224	6188			0.58					
		3	6016	6006			0.17					
		4	6562	6552			0.15					
		5	6552	6552			0.00					
25	**6**	1	4706	4680			0.56		4.18	14.03		
		2	5040	4420			14.03					
		3	4330	4290			0.93					
		4	4781	4680			2.16					
		5	4830	4680			3.21					
25	**8**	1	4239	3744			13.22		20.73	36.03		
		2	4810	3536			36.03					
		3	3882	3432			13.11					
		4	4410	3744			17.79					
		5	4624	3744			23.50					

Table F.11 (continued): Computational results for *General Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
40	**2**	1	30240	30240			0.00		0.00	0.00		
		2	28560	28560			0.00					
		3	27720	27720			0.00					
		4	30240	30240			0.00					
		5	30240	30240			0.00					
40	**4**	1	15848	15840			0.05		0.09	0.21		
		2	14992	14960			0.21					
		3	14536	14520			0.11					
		4	15848	15840			0.05					
		5	15840	15840			0.00					
40	**6**	1	11140	11088			0.47		0.52	0.92		
		2	10568	10472			0.92					
		3	10228	10164			0.63					
		4	11152	11088			0.58					
		5	11088	11088			0.00					
40	**8**	1	8720	8640			0.93		7.32	10.71		
		2	9034	8160			10.71					
		3	8522	7920			7.60					
		4	9484	8640			9.77					
		5	9295	8640			7.58					

Table F.11 (continued): Computational results for *General Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
55	**2**	1	56448	56448			0.00		0.00	0.00		
		2	53312	53312			0.00					
		3	51744	51744			0.00					
		4	56448	56448			0.00					
		5	56448	56448			0.00					
55	**4**	1	29254	29232			0.08		0.07	0.16		
		2	27652	27608			0.16					
		3	26810	26796			0.05					
		4	29246	29232			0.05					
		5	29232	29232			0.00					
55	**6**	1	20274	20160			0.57		0.46	0.79		
		2	19190	19040			0.79					
		3	18556	18480			0.41					
		4	20267	20160			0.53					
		5	20160	20160			0.00					
55	**8**	1	15996	15624			2.38		3.64	8.18		
		2	15963	14756			8.18					
		3	14596	14322			1.91					
		4	16342	15624			4.60					
		5	15800	15624			1.13					

Table F.11 (continued): Computational results for *General Problem* where $p_j \in [1,35]$ and $d_i \in [32,38]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
10	**2**	1	2400	2400	2400		0.00	0.00	0.00	0.00	0.00	0.00
		2	1860	1860	1860		0.00	0.00				
		3	1680	1680	1680		0.00	0.00				
		4	2220	2220	2220		0.00	0.00				
		5	2340	2340	2340		0.00	0.00				
10	**4**	1	1440	1440	1440		0.00	0.00	2.24	6.45	0.00	0.00
		2	1188	1116	1188	1693.10	6.45	0.00				
		3	1056	1008	1056	1279.40	4.76	0.00				
		4	1332	1332	1332		0.00	0.00				
		5	1404	1404	1404		0.00	0.00				
10	**6**	1	1245	1120	1240	621.22	11.16	0.40	24.57	49.59	0.08	0.40
		2	1188	868	1188	2190.65	36.87	0.00				
		3	1012	784	1012	1389.17	29.08	0.00				
		4	1265	1036	1265	1495.26	22.10	0.00				
		5	1350	1092	1350	1238.59	23.63	0.00				
10	**8**	1	1245	960	1240	621.22	29.69	0.40	45.33	59.68	0.08	0.40
		2	1188	744	1188	2457.44	59.68	0.00				
		3	1012	672	1012	1326.71	50.60	0.00				
		4	1265	888	1265	1495.26	42.45	0.00				
		5	1350	936	1350	1238.59	44.23	0.00				

Table F.12: Computational results for *General Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
25	**2**	1	13520	13520			0.00		0.00	0.00		
		2	10478	10478			0.00					
		3	9464	9464			0.00					
		4	12506	12506			0.00					
		5	13182	13182			0.00					
25	**4**	1	7280	7280			0.00		0.60	2.22		
		2	5767	5642			2.22					
		3	5136	5096			0.78					
		4	6734	6734			0.00					
		5	7098	7098			0.00					
25	**6**	1	5252	5200			1.00		7.20	22.13		
		2	4922	4030			22.13					
		3	3955	3640			8.65					
		4	4964	4810			3.20					
		5	5120	5070			0.99					
25	**8**	1	4553	4160			9.45		26.16	52.45		
		2	4915	3224			52.45					
		3	3739	2912			28.40					
		4	4582	3848			19.07					
		5	4926	4056			21.45					

Table F.12 (continued): Computational results for *General Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
40	**2**	1	33600	33600			0.00		0.00	0.00		
		2	26040	26040			0.00					
		3	23520	23520			0.00					
		4	31080	31080			0.00					
		5	32760	32760			0.00					
40	**4**	1	17600	17600			0.00		0.51	1.30		
		2	13784	13640			1.06					
		3	12480	12320			1.30					
		4	16312	16280			0.20					
		5	17160	17160			0.00					
40	**6**	1	12384	12320			0.52		3.03	6.89		
		2	10074	9548			5.51					
		3	9218	8624			6.89					
		4	11652	11396			2.25					
		5	12012	12012			0.00					
40	**8**	1	9952	9600			3.67		13.67	23.62		
		2	8952	7440			20.32					
		3	8307	6720			23.62					
		4	10116	8880			13.92					
		5	10000	9360			6.84					

Table F.12 (continued): Computational results for *General Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$

n	K	Ins. #	GH	Lower Bound	Optimal	CPU Times (sec)	LB Gap (%)	Optimality Gap (%)	Average LB Gap (%)	Maximum LB Gap (%)	Average Optimality Gap (%)	Maximum Optimality Gap (%)
55	**2**	1	62720	62720			0.00		0.00	0.00		
		2	48608	48608			0.00					
		3	43904	43904			0.00					
		4	58016	58016			0.00					
		5	61152	61152			0.00					
55	**4**	1	32480	32480			0.00		0.31	0.77		
		2	25352	25172			0.72					
		3	22912	22736			0.77					
		4	30068	30044			0.08					
		5	31668	31668			0.00					
55	**6**	1	22552	22400			0.68		2.00	4.11		
		2	18073	17360			4.11					
		3	16192	15680			3.27					
		4	21120	20720			1.93					
		5	21840	21840			0.00					
55	**8**	1	17698	17360			1.95		7.24	15.31		
		2	15514	13454			15.31					
		3	13424	12152			10.47					
		4	17275	16058			7.58					
		5	17076	16926			0.89					

Table F.12 (continued): Computational results for *General Problem* where $p_j \in [1,35]$ and $d_i \in [22,48]$