ABSTRACT

Title of Document:	AN ALGORITHM FOR CREW SCHEDULING PROBLEM WITH BIN PACKING FEATURES
	Wenxin Qiao, Master of Science, 2008
Directed By:	Ali Haghani, Professor and Chairman, Department of Civil and Environmental Engineering

This thesis proposes a new approach for solving the traditional crew scheduling problem. The crew scheduling problem is solved with a bin packing approach in polynomial time. Based on the extensive research on the bin packing problem during the past 40 years, an algorithm that is proved to be the most efficient for solving most bin packing problems is selected and modified for application in the crew scheduling problem. A Modified Best-Fit-Decreasing Algorithm is proposed and discussed in this study. A case study is conducted using the proposed algorithm and the results are discussed.

AN ALGORITHM FOR CREW SCHEDULING PROBLEM WITH BIN PACKING FEATURES

By

Wenxin Qiao

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Master of Science
2008

Advisory Committee:
Professor Ali Haghani, Chair
Paul Schonfeld
Cinzia Cirillo

© Copyright by
Wenxin Qiao
2008

Acknowledgements

Throughout the time I was writing this thesis, I received help from so many people who deserve my special acknowledgment.

First, I am very much indebted to my advisor, Professor Ali Haghani.It was his strong and continuous support over the past two years that guaranteed the completion of my thesis. Every part of my work is influenced by his wise directions and his patience in reviewing different versions of my thesis is really appreciated. His thoughtful and wise advices always directed me move on the right track for my study and I cherish everything I learned from him when facing obstacles and opportunities. Especially, I would like to thank him for giving me the courage to embrace the challenges and to defend what I believe in.

Second, I would like to show my deep appreciation to Professor Paul Schonfeld and Professor Cinzia Cirillo who served on my thesis committee. I want to thank them for contributing their precious time and expertise to better my work.

I feel grateful as well to my dear colleagues and friends for being there through my ups and downs. I would like to take this opportunity to give my special thanks to Masoud and Kaveh, who always gave me sound suggestions and advice on my research without any hesitation. I really appreciate their help and effort whenever I met with difficulties. I also want to thank Rafael, Abbas and other members in our group. The creative and cooperative atmosphere provided me the motivation in every step of the way. I feel lucky and thankful for being a member of this group. Although I may not have named everyone who contributed to my studies at the University of Maryland, they will always be on my mind and my deepest appreciation goes to those who are not named here but played a positive role in my life for the past two years.

Last, but not the least, I would like to dedicate this thesis to my parents whose love and support have kept me moving forward throughout my life.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

In this chapter, the background of the study undertaken in this thesis is discussed. The chapter describes the importance of the crew scheduling problem in transit system scheduling. The crew scheduling problem is defined here as well as the problem decomposition. Also, the research objectives and scope are given in this chapter.

Study Background

This section presents a brief introduction to the transit bus system, the scheduling process of the transit system, and the importance of crew scheduling is described here.

The Transit Bus System

The transit system has become an essential part of our daily life, especially in large developed urban cities. It is been a supporting industry for the nation to meet the goals of improving mobility, protecting the environment and saving energy. The transit system includes a variety of transportation modes such as bus, railway, and ferry. Among these, the bus system plays a very critical role as it serves the largest number of customers in the transit system. A transit bus company uses two key resources for its service: vehicles and crews, and the object of this study is crew scheduling for the bus company.

Many cities around the world are investing in public transport to increase its attractiveness and usage. Bus priority is encouraged in most countries. First, the bus system can greatly help reduce serious congestion in the network as the congestion cost is huge. By attracting a larger ridership on the bus, fewer private cars will appear on the road and this will help a lot to relieve the heavy traffic congestion, especially during the morning and afternoon peak hours. Second, the public bus system can help by saving a large amount of energy resources, such as gas. Less fuel will be consumed if more people are willing to take the bus system instead of using private cars. This is more environmentally friendly since many transit companies are adopting the new clean fuel buses to reduce air pollution.

The Scheduling Process of the Transit Bus System

Developing a convenient and efficient bus system depends heavily on an efficient and fast scheduling process of the bus system. The transit bus scheduling process usually includes 4 steps: Designing Routes, Setting Frequencies and Building Timetables, Vehicle Scheduling, and Crew Scheduling (Figure 1).

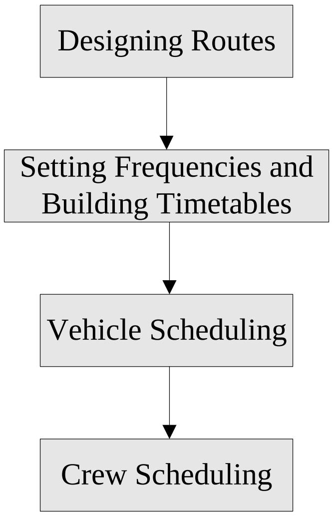


Figure 1 Scheduling process of transit bus system

In the stage of Route Design, according to the service demand and the feasible links in the network, a set of routes are selected and designed to serve the target area. In the second stage, based on the available vehicle resources, budget, demand and the level of service required, the schedulers set the frequencies and build the timetables according to the trip information. The output of this process is trip information with their corresponding starting and ending time and locations. In the third stage, vehicle scheduling, the vehicles are assigned to the trips. Each vehicle is allocated to a route to serve and a schedule is set for each vehicle starting and returning to the depot after serving a sequence of trips. And usually, the goal of vehicle scheduling is to find the minimum number of vehicles needed to meet the level of service required. Crew scheduling is in the last stage in the whole scheduling process. During this process, the tasks are assigned to the bus drivers. The crew runs are generated at this stage, which is one day’s work for one bus driver. Many constraints are taken into consideration to obtain feasible and legal runs, including both labor rules and company rules.

This scheduling process is usually carried out twice a year in a transit company. This process is both labor intensive and time consuming since it includes collecting the data, and scheduling vehicles and crews. It takes the schedulers a significant amount of time to come up with a new schedule and it is impossible to change the current schedule on a short notice. As a result, computerized systems for the transit scheduling have been adopted in most bus companies to facilitate this complicated process and function in a more cost effective way.

The Importance of Crew Scheduling

Crew costs take up a great part in the total operation cost for a transit bus company. From the company's perspective, the largest single cost item is the driver's wage and fringe benefits. Crew cost can take up to 80% of the total operation cost (Bodin et al. 1983) which dominates other costs such as the vehicle costs and maintenance fees. Crew scheduling is the last but an important part in the transit scheduling process as minimizing crew cost will largely help optimizing the total operational costs for the bus company.

It is very important for the process of crew scheduling to be both efficient and fast, and computer aided scheduling have a great advantage in this. Computerized crew scheduling has been studied for decades and is gradually taking the place of the original manual scheduling. The change from traditional manual scheduling to computer aided crew scheduling system is based on several reasons discussed below.

The first reason is that the public transit market has become more competitive than before with economic growth. To gain more benefits while maintaining the level of service required, more attention is paid to reducing the operational cost of the transit company. Research results have shown that computer aided scheduling can help save 1% to 3% of the total costs (Bodin et al. 1983) compared to manual scheduling, however, this small portion can still add up to a large amount of money for a large bus company.

Secondly, the transit system in the US is expanding its service area all the time. Transit priority is strongly encouraged by the government to reduce congestion and protect the environment. Many suburban areas have become urbanized cities over the past years. As a result, the scheduling problem we are facing has become more large-scaled and complicated. On one hand, the extensive size and the complexity of the problem made manual scheduling cumbersome and slow. On the other hand, with the increased speed of the computers and more developed solution algorithms, computerized scheduling has shown an obvious advantage over the traditional manual scheduling both in the processing time and cost efficiency.

Finally, the bus schedule is not fixed once it is set, and it usually changes once or twice a year according to changes in the service demand. It will take a long time for the schedulers to make a new schedule manually and it takes years for training a new scheduler into an experienced one. In addition, a computerized scheduling system adds a lot of flexibility in the whole scheduling process. With a computerized scheduling system, it is possible to determine the crew cost as a function of changes in parameters that impact it such as work rules and route choice. It provides a way for the schedulers to evaluate the changes in the system quickly and accurately which will support the decision making for the company.

The Crew Scheduling Problem

The crew scheduling problem has been extensively studied in the operation research literature. For many years, a common used method for crew scheduling was manual scheduling by the experienced schedulers and it is still used in some bus companies. Over the past 30 years, much research has been done on computer-aided transit scheduling. Computerized scheduling systems have been used in crew scheduling and are becoming widely used by transit agencies. Since the first international workshop on computer-aided transit scheduling held in Chicago in 1975, 10 international workshops have been held on this topic during the past 30 years. Many great advances and major contributions, as well as the most state-of-the-art computer aided transit scheduling techniques have been presented in these workshops. These workshops were held in turn among North American, Europe and Canada and are listed as follows: Chicago (1975), Leeds (1980), Montreal (1983), Hamburg (1987), Montreal (1990), Lisbon (1993), Massachusetts (1997), Berlin (2000), San Diego (2004), and Leeds (2006).

The definition of the crew scheduling problem

The crew scheduling problem (CSP) is a process of assignment that given a bus schedule, cut the vehicle blocks into pieces of work and combine these pieces into legal and feasible runs to assign to the bus drivers, provided that all tasks are covered by the runs with the objective that the total crew cost is minimized. The CSP is very challenging since it covers both spatial and temporal issues, which consider both the movements of the vehicles to cover the blocks and a variety of time constraints on the crew runs based on labor and company rules.

Definition of terms

A large set of terms are used in the crew scheduling problem and there is large variation in the same terms across different countries. The definitions of several important terms used in this thesis are listed below. For a more comprehensive list of defined terms, please refer to the glossary in appendix.

Block: A sequence of tasks assigned to one bus for one day's work; a block is a vehicle assignment;
Run: The work performed by a single crew for one single day; a run is a crew assignment;
Task: The trip between two relief points on a block; a task is part of a piece of work.
Piece of work: A piece of work is a continuous work period composed by one or more consecutive tasks covered by the same crew; a piece of work is a part of a run.
Relief Point: The stops along the route where a crew can take a meal break and another crew can take over the bus; a relief point is a location where one crew can replace another.
Depot: Parking and service location for vehicles when is not required for service.

Partition: A partition of a block is the selection of a set of cuts each representing a relief point.

Block Partition: A set of pieces of work which covers exactly the block.

Figure 2 illustrates the relationship between task, piece of work and block.

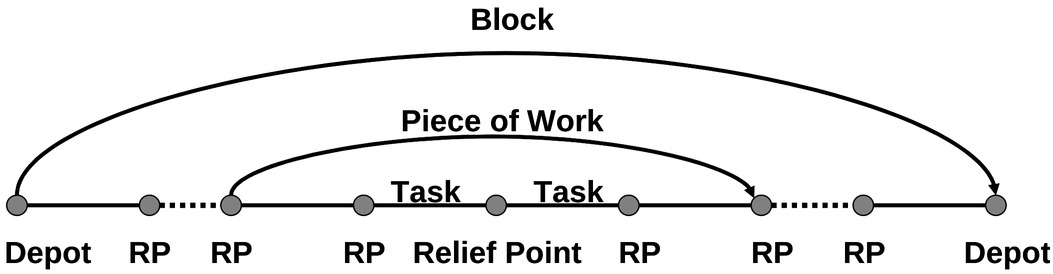


Figure 2 Relationship between task, piece of work and block

For a complete definition of terms used in crew scheduling, please refer to Hartley's a glossary of terms in bus and crew scheduling, which also includes the variations used in different countries (Hartley, 1981).

Problem decomposition for crew scheduling

The proposed algorithms for the crew scheduling problem usually solve a sequence of two or more subproblems where the output of the first subproblem is the input of the second subproblem and the output of the second is the input of the third and so on. The nice feature of this decomposition is that each subproblem contains relatively less data and may be solved with an efficient solution algorithm.

Many different decomposition methods are used in the CSP. Some divide the problem in three steps: partition of the timetable into blocks, graph generation and run achievement (Blais and Rousseau 1987); some solve the problem in two parts: formulate the block partitioning into a shortest path problem and then solve the run generating as a matching problem (Ball et al. 1983).

In this thesis, the crew scheduling process is decomposed into two parts: block cutting and piece matching. In the block cutting process, all vehicle blocks are cut into pieces of work which generate an initial candidate set of pieces to be combined into runs in the second step. Many different partitions can be cut for the same block and for this study all possible partitions are generated. During the piece matching process, these pieces of work from the candidate set are matched and combined into feasible and legal runs. The matching criteria are both spatial and temporal. First, the matching pieces should be geographically compatible, which means the drivers are able to perform a changeover at the relief point. In this study, it is considered that the connecting two pieces share the same relief point at the cut. Second, the starting time of the second piece should be no earlier than the ending of the first piece and all the time constraints related to labor rules and company rules need to be satisfied, which is the complicated part. Also, the goal of the matching is that all blocks are covered at a minimum cost. The whole process is shown in Figure 3.

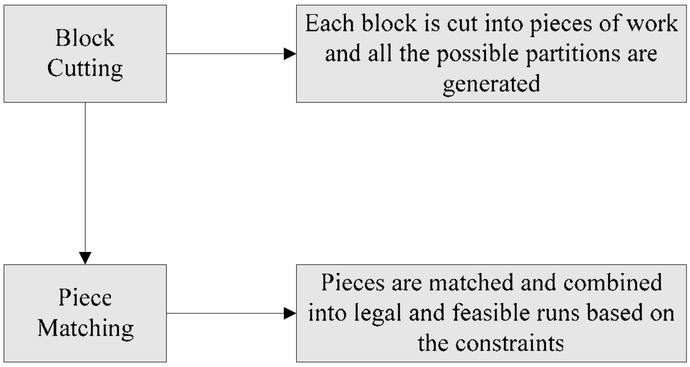


Figure 3 Problem decomposition description

Research Objectives and Scope

The research objectives and the scope of the study is described in this section.

Research Objectives

The objectives of this research are to propose a new algorithm for the CSP, solve test problems using this algorithm; conduct a case study and solve the problem with the methodology proposed. To achieve these objectives, a comprehensive literature review of the crew scheduling models and algorithms that are developed over the past 30 years as well as models and algorithms developed for the bin packing problem is conducted and the crew scheduling problem studied in this thesis is clearly defined.

Research Scope

In this thesis we study a basic version of the crew scheduling problem which is given the vehicle blocks, solve the CSP with an output of the crew runs and total cost. The crew working rules and the feasibility of each run are considered here. Some of the crew working rules considered include: the maximum spreadover time for a run, the maximum working time, the minimum and maximum piece length, and a minimum meal break time between the pieces. The company rules that require a maximum or minimum percentage of each different type of run are not taken into account in this study. Although this is a basic version of the CSP, it is still an NP hard problem. The objective is to minimize the total cost of the runs with all the tasks covered where the cost is consisted of the crew working hours and the spread over penalty. For the case study, first, several small problems are tested with the proposed algorithm. Then, a case study with 73 vehicle blocks and 1299 tasks is solved using this methodology. The labor rules and cost function are assumed to be the same as those in the original problem.

Overview of the Thesis

This thesis is organized as follows:

Chapter 1: Introduction

In this chapter, the background of this study is introduced, including a description of the transit bus system as well as a general overview of the scheduling process of the transit system. The importance of the crew scheduling problem, which is the last step in the transit scheduling process, is emphasized with a description of the advantages of computer aided crew scheduling system. The definition of the crew scheduling problem is given in this chapter along with the definition of terms. Also, the decomposition of the problem is briefly discussed in this chapter. Last, the research objectives and scope are defined.

Chapter 2: Literature Review

In this chapter, a literature review is conducted to help us understand the general approaches to the crew scheduling problem over the past three decades. Of those proposed algorithms, three types of common approaches are described in detail: the run cutting heuristic, matching algorithm and set covering approach. The proposed algorithm in this thesis adopts the idea of bin packing and a literature review of the bin packing problem is also conducted in this chapter, including its concept, algorithms and a brief analysis of the worst case performances of these bin packing algorithms. In addition, the application of bin packing in the CSP is introduced.

Chapter 3: Crew scheduling problem statement and solution algorithm

This chapter states the crew scheduling problem studied in this thesis. The definition, mathematical model and assumptions are described for the problem. The solution process of the proposed approach for the CSP is explained in detail. The block cutting process and piece matching process are both introduced with a process overview as well as a description of the solution method. At the end of this chapter, several test problems are studied using the proposed approach.

Chapter 4: Case study-solving the crew scheduling problem for a transit bus system

In this chapter, a case study is conducted. With a description of the study overview, its objectives and scope, as well as the analysis of data provided, the problem is solved with the proposed approach and a detailed description of the solution is given for both the block cutting and piece matching process.

Chapter 5: Conclusions and direction for future study

This chapter gives a summary and conclusion of the study. Recommendations for future studies are also discussed in this chapter.

Chapter 2: Literature Review

For the past 30 years, the crew scheduling problem has been extensively studied in and out of the transportation literature. A large amount of relevant papers on this topic can be found in the journals of operation research, mathematics, computers and management science. In literature, the Crew Scheduling Problem (CSP) is generally formulated as a zero-one integer linear programming problem. This problem is very complicated and proven to be NP-hard in both single depot and multiple depot cases (Fischetti et al. 1989).

General Approaches to Solve the CSP

An abundance of mathematical and heuristic approaches to this problem were proposed during the past three decades. Of the variety of these approaches to the CSP, three main approaches are classified and reported in the literature (Bodin et al. 1983, Wren and Rousseau 1995): the run cutting heuristic, the matching algorithm, and the set covering approach. An introduction of the research work for each of these three categories is provided in this section.

Run Cutting Heuristics

This category follows the technique used by the manual schedulers, which is called run cutting. The run cutting heuristic was firstly used in the 1970s as a constructive algorithm that emulates the procedures used by manual schedulers. It is used in the American RUCUS package and the British TRACS system. This algorithm cuts the uncovered pieces from a block and covers the pieces with a run. The run cutting heuristics are carried out in two steps. The first step cuts the vehicle blocks into pieces of work and combines two or three pieces to form a feasible run. During this process a piece of work is covered if a run is assigned to it and is uncovered otherwise. The process repeats until all pieces are covered with runs. In the second step, the cost of runs is improved by modifying the cutting of a vehicle block into pieces of work to obtain better pieces or by exchanging pieces between the runs.

L.Cavique et al. (1999) presented two alternative improvement algorithms embedded in a tabu search framework to reduce the number of runs generated from the initial solution by the traditional run cutting approach. First, an initial solution is generated by the run cutting approach, then in the improvement stage, Tabu-crew takes advantage of using strategic oscillation in searching for new solutions and the Run-ejection algorithm is based on a sub graph ejection chain method which considers compound moves. Computational results are given for a real world case of 21 blocks with 528 trips and 79 runs.

Matching Algorithms

The matching algorithm is used in RUCUS-II and an initial version of HASTUS (Blais and Rousseau 1987). This method is divided into three parts: partition of the blocks into pieces of work, matching graph generation, and run achievement. Initially, the blocks are partitioned into pieces of work, then the piece graph is generated for solving the matching problems, and finally runs are generated by matching the pieces. This process is formulated as a matching problem. In HASTUS, the minimum cost matching problem in arbitrary (non-bipartite) graphs is based on the solution of a succession of minimum cost network flow problems.

Ball and Bodin (1983) proposed a matching based heuristic to schedule crews and vehicles simultaneously. Their algorithm is decomposed into three components: piece construction component, piece improvement component and run generation component. Several sub problems are formulated and solved as matching problems on graphs. Computational results are given for a real world problem with 1602 tasks and 139 crews. S. Martello and P. Toth (1986) proposed a heuristic approach by using a greedy procedure that builds upon a partial solution, guided by the solution to a number of matching problems. Ball and Benoit-Thompson (1987) decomposed the CSP into a shortest path problem and a matching problem. The block partitioning problem is formulated as the shortest path problem where blocks are cut into pieces. The run generation problem is formulated as a matching problem where each run consists of either one or two pieces. This approach iterates between the solutions of the shortest path problem and the matching problem.

Set Covering Approach

This category formulates the CSP as a set covering problem (SCP) or set partitioning problem (SPP), in which the approaches are based on set covering or set partitioning models. The set covering approach is used in the package IMPACS of the BUSMAN system and in the Crew-Opt package. This method first generates a large set of feasible runs and then finds the minimal covering set. The algorithm usually has two embedded loops where the internal loop creates a candidate list of all feasible runs, while the external loop selects the best runs from the candidate list. Most approaches solved this problem by linear programming or Lagrangean relaxation and then applied a branch and bound algorithm (Heurgon 1975, Shepardson and Marsten 1980, Wren et al. 1985).

The crew scheduling problem can be illustrated by rows and columns in the SCP where the rows are the pieces of work and the columns are the runs. The goal is to get a minimum cost set of columns with each row included in one of the columns in the solution. The mathematical model for the SCP and SPP is shown as below.

Set covering model:

$$\text{Min: } \sum_{j=1}^{n} c_j x_j \quad j = 1......n$$

$$\text{s.t. } \sum_{j=1}^{n} a_{ij} x_j \geq 1 \quad i = 1......m$$

where c_j is the cost of run j

x_j is the jth run

$$x_j = \begin{cases} 1 & \text{if } j\text{th run is in the solution;} \\ 0 & \text{otherwise} \end{cases}$$

$$a_{ij} = \begin{cases} 1 & \text{if } i\text{th piece is in the } j\text{th run;} \\ 0 & \text{otherwise} \end{cases}$$

Set partitioning model:

$$\text{Min: } \sum_{j=1}^{n} c_j x_j \quad j = 1......n$$

$$\text{s.t. } \quad \sum_{j=1}^{n} a_{ij} x_j = 1 \quad i = 1......m$$

$$\text{where } x_j = \begin{cases} 1 & \text{if } j\text{th run is in the solution;} \\ 0 & \text{otherwise} \end{cases}$$

$$a_{ij} = \begin{cases} 1 & \text{if } i\text{th piece is in the } j\text{th run;} \\ 0 & \text{otherwise} \end{cases}$$

For the SPP, the objective is to select a set of feasible runs with minimum cost that each piece of work is covered exactly once in one run. To relax the set partitioning problem, set covering model is usually used by replacing the equalities $\sum_{j=1}^{n} a_{ij} x_j = 1$ in SPP with the inequalities $\sum_{j=1}^{n} a_{ij} x_j \geq 1$. In the set covering formulation, over covering is allowed which is not a problem in practice, since the overlaps can be manually deleted and it can only happen when it is less expensive to assign one task to two crews instead of one crew.

Mitra and Darby-Dowman (1985) proposed a set covering solution for the CSP by using integer linear programming (ILP). Falkner and Ryan (1987) solved the CSP as a set partitioning problem. Smith and Wren (1988) also solved the CSP with the set covering method using ILP. Their covering process uses slack and surplus variables. Fischetti et al. (2001) proposed a 0-1 linear programming formulation based on binary variables in a simplified crew scheduling case in which they impose limits on both the spread time and working time. Their model is enhanced by new families of valid inequalities and the method is embedded in an exact branch and cut algorithm. A feature of their formulation is that the LP relaxation is solvable in polynomial time. Computational results are given for random and real world test cases with between 50-500 trips and 27-84 crew members.

Two methods are commonly used to cut the computation time of the SCP/SPP: the Lagrangian relaxation method and the column generation technique. A combination of these two techniques is also studied in the literature.

The idea of Lagrangian relaxation is to relax some of the difficult constraints and penalize their violations by certain weight in the objective function. It is used to obtain lower bounds on the optimal solution for a combinatorial optimization problem. This technique can be applied to the general integer programming problem, where the complicated constraint is removed and weighted by a given Lagrange multiplier and put into the objective function. For example, the following model has the set of constraints split into two sets, hard and easy constraints (a) and (b) respectively.

$$\text{Min: } \sum_{j=1}^{n} c_j x_j \quad j = 1......n$$

$$\text{s.t.} \quad \sum_{j=1}^{n} a_{ij} x_j = 1 \quad i = 1......m \qquad \text{(a)}$$

$$\sum_{j=1}^{n} b_{kj} x_j = d \quad k = 1......p \qquad \text{(b)}$$

$$x_j \in \{0,1\}$$

A Lagrangian multiplier λ_i is introduced for the hard constraint (a). This constraint is removed from the model and penalized in the objective function. The Lagrangian subproblem is formulated as below where $\phi(\lambda)$ is the lower bound of the optimal solution to the original problem:

$$\phi(\lambda) = \min \sum_{j=1}^{n} c_j x_j - \sum \lambda_i \left(\sum_{j=1}^{n} a_{ij} x_j - 1\right)$$

$$\text{s.t.} \quad \sum_{j=1}^{n} b_{kj} x_j = d \quad k = 1......p \qquad \text{(b)}$$

$$x_j \in \{0,1\}$$

Beasley and Cao (1996) present a zero-one integer linear programming formulation for a generic crew scheduling problem, which is based upon Lagrangean relaxation to provide a lower bound for imbedding in a tree search (branch and bound) procedure together with subgradient optimization to solve the problem optimally. Computational results are given for random test cases involving between 50-500 tasks with 27-204 crew members.

The second method is the column generation technique. This technique is adopted when the number of variables become huge. The basic idea of column generation is introduced by Dantzig and Wolfe (1960). It solves a sequence of reduced subproblems, and each subproblem contains only a small subset of the variables (columns). When a subproblem is solved, a new set of columns is obtained by using dual information from the solution. Instead of repeatedly scanning a large number of columns (runs), only subsets of columns are selected. If the optimal solution is not found, the corresponding subproblems will be solved. The column generation algorithm will converge once the optimal solution based on the current set of columns can not be improved by adding more columns. At this point, the optimal solution of the subproblem is the optimal solution of the original problem.

Desrochers and Soumis (1989) solved the problem as a set covering problem using column generation method based on a shortest path algorithm, rather than ILP. They decomposed the problem into a set covering problem and a subproblem. In the first part, which is a linear relaxation of the set covering problem embedded in a branch and bound scheme, a schedule is chosen from already known feasible runs. In the second subproblem, which is defined as a special case of the shortest path problem with resource constraints on an acyclic graph, a new feasible run is generated to improve the current solution. Their method can produce a good valid lower bound on the solution cost and is applicable to multiple-piece runs. Computational results are given for two real world cases: one with 25 blocks, 167 tasks and 45 runs, the other with 20 blocks, 235 tasks and 51 runs.

Carraresi et al. (1995) proposed a column generation approach that starts with a feasible set of runs and iteratively replaces some runs to get better solution, and the solution uses a Lagrangean relaxation method. Gaffi and Nonato (1997) proposed an integrated approach for ex-urban vehicle and crew scheduling. They use the column generation method to solve the set covering or set partitioning continuous relaxation embedded in branch and bound framework to get an integer solution. Huisman et al. (2005) presented two algorithms for integrated vehicle and crew scheduling in multiple-depot case based on a combination of column generation and Lagrangian relaxation. Computational results are given for real world cases involving between 194-653 trips with 28-117 crew members.

Other heuristics are also proposed for solving the CSP more efficiently. Genetic algorithms and tabu search are examples of the meta-heuristics used for the crew scheduling problem. Clement and Wren (1995) solved the CSP using a genetic algorithm in which feasible crew schedules were encoded as chromosomes. The search for optimal solution is based on rules similar to genetic survival mechanism and greedy algorithms are used. Lourenco et al. (2001) solved a multi-objective CSP since in practice, several conflicting objectives should be considered when determining a crew schedule. They use a tabu-search technique and genetic algorithms to solve the problem.

Other innovative approaches to the CSP include the work of Banihashemi and Haghani (2001) who formulate the CSP as a task-based multi-commodity network flow problem, where the variables are defined in conjunction with the tasks and task compatibilities. This approach starts from an initial relaxed problem, considering minimum costs for these compatibilities. Then an iterative procedure is proposed for building the runs and adjusting the compatibility costs when necessary. New variables are generated corresponding to the established feasible runs and a soft constraint is associated with each new variable. Paixao (1990) formulated the SCP using dynamic programming. The search process uses a state-space relaxation method. The later work in Paias and Paixao (1993) extended this approach by showing that the lower bound solution is found.
Li and Kwan (2003) proposed a hybrid genetic algorithm for the bi-objective CSP. A greedy heuristic is used to construct a schedule by sequentially selecting runs from a large set of pre-generated potential runs to cover the remaining work. Fuzzy set theory is applied in the evaluation of individual runs and the schedule as a whole.

General Steps to solve the CSP

A variety of approaches have been proposed in the CSP literature. In general, the procedure of solving the crew scheduling problem can be basically summarized as in the following 3 steps:

First step: Generate an initial candidate set of feasible solutions—usually generated by using a run cutting heuristic. Column generation technique is often used to expand the candidate set.

Second step: Select from the candidate set a minimum cost subset to cover all the tasks—usually the problem is formulated as a set covering or set partitioning problem.

Third step: Solve the SCP or SPP using appropriate algorithms— usually solved by linear programming or Lagrangean relaxation with a branch and bound scheme.

Figure 4 shows a flow chart with the three steps.

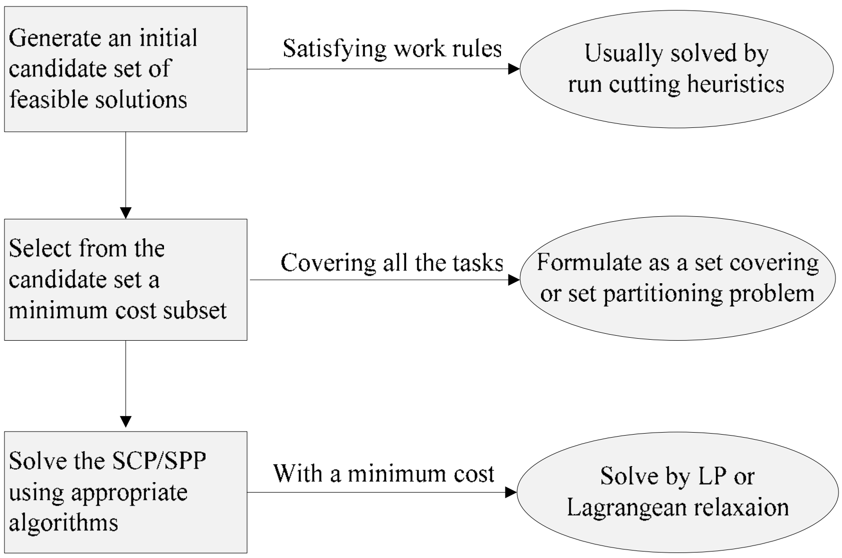


Figure 4 General steps to solve the CSP

Integrated vehicle and crew scheduling has also been proposed and studied for years. Some relevant work are: Ball et al. (1983), Darby-Dowman et al. (1988), Gaffi and Nonato (1999), Haase et al. (2001) and Freling et al. (1999, 2003). In this thesis, the study follows the pattern of a sequential approach, i.e. given the results from the vehicle scheduling, solve the crew scheduling problem independently.

A Bin Packing Approach to Solve CSP

In the thesis, bin packing is proposed as a new approach for solving the matching part of the CSP. The piece matching process is formulated as a bin packing problem (BP) and the algorithm used for solving the BP is modified and applied to the CSP. A review of the bin packing problem literature and its several common solution algorithms is given below.

The Bin Packing Problem

The Bin packing problem has been extensively studied as a combinatorial problem in the literature of operation research and mathematics. This problem is proven to be NP-hard by Coffman et al. (1976). As early as the 1960s Gilmore and Gomory had studied a cutting stock problem with a similar concept to BP. All known algorithms for finding the optimal solution to this problem require exponential time. Since the early 1970s, extensive research has been done to find the near optimal solution to the bin packing problem.

In literature, the bin packing problem is defined as: given a set of *n* items, each with a size between 0 and 1, and a set of empty bins, each with a capacity of 1, pack the *n* items into a minimum number of bins, provided that the sum of item sizes in each bin does not exceed 1.

The classic bin packing problem is called the one-dimensional bin packing problem (1BP), which is the study base for a variety of bin packing problems. In the 1BP, there is an unlimited supply of identical bins, each with a positive capacity *c*. Given a set of *n* items, each having a positive weight w_i, pack them into a minimum number of bins with the constraints that the sum of the item weights in each bin does not exceed the given capacity ($\sum_{i \in bin} w_i \leq c$). Let M denote the minimum number of bins needed, it can be shown that: $M \geq \left\lceil \frac{\sum_{i=1}^{n} w_i}{Capacity} \right\rceil$, and M$\leq$ number of bins required< 2M. If more than 2M bins are used, it means that there exist two bins with their combined item weights less than the capacity *c* thus could be combined into one bin.

The bin packing problem has many real world applications such as cutting stock units in the wood and glass industry, packing goods on shelves in warehouses, loading trucks subject to weight limitation, optimizing cable length, paging newspapers, as well as the application in the machine scheduling to minimize the number of machines needed for completing all the tasks within a certain time frame. The applications are discussed in Johnson (1974) and Coffman et al. (1976).

Heuristic Algorithms for the Bin Packing problem:

The bin packing problem is NP hard in the strong sense (Coffman et al. 1976). Since a polynomial time algorithm is not likely to be found for this problem, an enumeration method is not practical to get the optimal solution due to the prohibitive amount of computation time required. This problem is very difficult to solve in practice as no good mixed integer programming (MIP) formulation has been found so far for the packing problem.

Heuristic algorithms have been widely used to solve practical problems in operations research, because a large variety of OR problems are proved to be NP hard and exhaustive search method for the optimal solution is often computationally impossible. Thus, instead of solving the problem optimally, fast heuristic algorithms are provided to generate a good, near optimal solution. A recent comprehensive survey of the bin packing algorithms can be found in Coffman et al. (1996).

Bin packing problems can be classified in several ways. One classification is based on their dimension. There are one-dimensional, two-dimensional, three-dimensional and multi-dimensional bin packing problems. For the bin packing algorithms, two categories of algorithms are mainly studied: online and offline algorithms. For the online algorithms, items arrive in a given order and are assigned to bins at arrival without knowledge of the item yet to arrive. The commonly used online algorithms are First-Fit Algorithm and Best-Fit Algorithm. For the offline algorithms, items are sorted prior to their arrival. And First-Fit-Decreasing Algorithm and Best-Fit-Decreasing Algorithm are commonly used as offline algorithms.

The first and most fundamental heuristic for the BP is the Next-Fit algorithm (NF): an item is packed in the current bin if possible, or else a new bin is opened and becomes the current bin and the item is packed in there. Next-Fit-Decreasing (NFD) is the offline form of the NF algorithm where the items are sorted in a decreasing order beforehand. In the literature, there are four efficient algorithms commonly used which have the same order of time complexity, O (n log n), (Johnson et al. 1974):

1. First-Fit Algorithm: The items are taken successively and fit into the first bin (lowest indexed bin) in which it fits.
2. Best-Fit Algorithm: The items are taken successively and fit into the most nearly full bin (fullest bin) in which it fits.
3. First-Fit-Decreasing Algorithm: Same as the first fit algorithm except that the items are firstly sorted by size in decreasing order. Items are packed in order of size, largest first and put in the first bin where they will fit or on the bottom of a new bin if no such bin exists. A new bin is opened if necessary.
4. Best-Fit-Decreasing Algorithm: Same as the best fit rule except that the items are firstly sorted by size in decreasing order. BFD makes the algorithm a little more complicated then FFD, but usually no better.

In addition to the fast heuristic algorithms introduced above, Martello and Toth (1990) introduced the Martello-Toth Reduction Procedure (MRTP). Their approach repeatedly reduces the original problem to a simpler subproblem by exploring the optimal combination of 1-3 items before packing them into the bins. This may eventually stop with some items remaining unpacked and the remainder packed using a largest first, best fit algorithm.

It is usually difficult to evaluate and compare the performance of the heuristics. Originally, to evaluate the performance of the heuristic, the heuristic is programmed and tested on a representative sample problem which is empirical testing. However, testing them on a large problem set with known exact optimal solution is impossible. One way to evaluate the performance is to mathematically analyze the performance to see how close the heuristic solution is to the optimal solution. To evaluate the performance of the heuristics, three analytic approaches are usually used (Fisher 1980 and Ong et al. 1984): worst case analysis, probabilistic analysis and statistical analysis.

- The worst case analysis establishes the maximum deviation from optimality that can occur when a specified heuristic is applied within a given problem class.
- The probabilistic analysis assumes a density function for the data and establishes the heuristic probabilistic properties, such as a probability bound for the heuristic to find a solution within prespecified percentage of optimality.
- The statistical analysis usually applies the heuristic on a large number of sample problems to draw some statistical inference on the heuristic.

The worst case analysis for several bin packing heuristics was studied by Johnson et al. (1974) and Coffman et al. (1978). Let FF (n), BF (n), FFD (n), BFD (n) denote the number of bins used in applying each of the above algorithms to a set of *n* elements. Let OPT denote the optimal number of bins needed. The worst case performances for these heuristics are given in Johnson et al. (1974) as follows:

$$FF(n) \leq (17/10)OPT + 2;$$

$$BF(n) \leq (17/10)OPT + 2;$$

$$FFD(n) \leq (11/9)OPT + 4;$$

$$BFD(n) \leq (11/9)OPT + 4;$$

The absolute performance ratio for a heuristic gives the solution’s maximum deviation from optimality. The asymptotic worst-case ratio for a heuristic gives the solution’s maximum deviation from optimality for all lists that are sufficiently large. Let H(n) be the number of bins generated for packing n items by heuristic H and OPT (n) be the optimum number of bins required, $R_n[H] = \max\{\frac{H(n)}{OPT(n)} : OPT(n) = n\}$. The asymptotic worst-case ratio for heuristic H is:

$$R^{\infty}[H] = \lim_{n\to\infty} \sup R_n[H]$$

Table 1 below shows the asymptotic worst-case ratios for these heuristics for the classical one dimensional bin packing problem.

Table 1 Heuristics asymptotic worst-case ratios

Heuristic	$R^{\infty}(H)$	Reference
FF	1.70	Johnson et al.
FFD	1.22	Johnson et al.
BF	1.70	Johnson et al.
BFD	1.22	Johnson et al.
NF	2.00	Coffman et al.
NFD	1.69	Baker and Coffman

Not much research has been done on the probabilistic and statistical analysis of the algorithms. Explicit probabilistic results for these heuristics are difficult to get since the dependency between the successive packing operations is tightened. A statistical approach is applied to analyze the expected performance by Ong et al. (1984). Their study shows that the expected number of bins required by these heuristics can be approximated by a linear function in the form of $a + b$ n, where the constants a and b are estimated by the least squares method based on a computational simulation from 20 samples. The 500-point scatter diagram for the sample data and the regression line were plotted to show that the estimators are very good.

Crew scheduling problem has been studied for more than 30 years, and a variety of mathematical models and heuristic approaches for solving this problem have been proposed in the CSP literature. Among these, three main approaches were classified and described in this Chapter: run cutting heuristic, matching algorithm, and set covering approach. From the literature, we summarized the three general steps to solve the crew scheduling problems as: (1) generating an initial candidate set of feasible solutions,

(2) selecting from the candidate set a minimum cost subset to cover all the tasks, (3) solving the SCP or SPP using appropriate algorithms.

Heuristic algorithms are very popular in solving the NP hard crew scheduling problems since exhaustive search for the optimal solution is often computationally impossible, especially when the problem size grows. Review of the heuristics approaches proposed and applied in CSP indicates that solving CSP with a bin packing approach is an area that has not been studied and discussed before. Bin packing problem, as a combinatorial problem in literature of operation research and mathematics, has been discussed for nearly 40 years. Based on an extensive review of the literature in the original bin packing problem, the author believes that using an approach based on bin packing may hold promise in solving large scale CSP. The piece matching process in CSP can be formulated as a bin packing problem. Considering the similar nature of the BP and CSP, modifying and applying an efficient BP algorithm has potential for providing a faster heuristic to find better, near optimal solutions.

Chapter 3: Crew Scheduling Problem Statement and Solution Algorithm

Crew scheduling is a multi-objective task. During this process, the schedulers need to guarantee that all bus schedule are completly covered, all runs are feasible and legal, and the total operation cost is minimized. This chapter defines the crew scheduling problem studied in this thesis, its mathematical model as well as the proposed solution methodology. The proposed approach for solving the CSP that involves the block cutting process and piece matching process is explained in detail. At the end of this chapter, 3 test problems are solved using the proposed approach.

Problem Description

This thesis considers a basic version of the crew scheduling problem where the crew working rules and the feasibility of each run are considered. Some of the crew working rules considered are the maximum spreadover time for a run, the maximum working time, and the minimum meal break time. The working time is the spreadover time of a run minus the meal break time. The objective is to minimize the total cost of the runs. Although this is a basic version of the CSP, it is still an NP hard problem.

Problem definition

The original crew scheduling problem is formulated as: Given a set of *m* blocks, cut the blocks into pieces of work and cover all the pieces of work by *n* runs with a minimum total cost, provided that each run complies with the labor rules laid down by the union contracts and the company regulations are satisfied. Different approaches for solving this problem were described in chapter 2.

In this study, the crew scheduling problem is formulated with a bin packing feature in the following way. Given a set of *m* blocks, first cut each blocks into different partitions with each partition consisting of several pieces of work. All possible partitions satisfying the constraints are generated in this process. Second, pack these pieces into a set of feasible bins (crew runs) with a minimum total cost, provided that no two pieces overlap and the constraints for labor rules are satisfied.

These assumed working rules are mainly based on the TRB manual- TCRP report 30: Transit Scheduling: Basic and Advanced Manuals. The assumptions made for this problem are: given the blocks, the bins (runs) are generated subject to the following constraints:

1) The maximum total working time for a run is 9 hours;
2) The maximum spread over time for a run is 12 hours;
3) The minimum meal break time is 30 minutes.
4) A meal break must exist after a maximum working period of 6 hours.
5) A piece of work has a minimum length of 2 hours and a maximum of 6 hours.
6) The guaranteed paid time for each run is 8 hours.
7) The crews take a break only at the relief points and the change over of buses is at the same relief point.

Mathematical Model

The objective of this problem is to minimize the total costs of the bins (runs) with all tasks covered by the runs. The mathematical model for the problem is formulated as below:

Define a binary decision variable x_j, $j = 1,2,...,n$, where x_j is the *j*th run. The cost for each run depends on its total working time and spreadover time. Spreadover penalties and the guaranteed paid hours are considered in the cost function. The cost function for the runs is set as:

$$C_j = \begin{cases} c_1 * W^g & \text{if } W_j < W^g, \quad S_j < S_{\max}; \\ c_1 * W_j & \text{if } W_j > W^g, \quad S_j < S_{\max}; \\ c_1 * W^g + c_2 * (S_j - W_{\max}) & \text{if } W_j < W^g, \quad S_j > S_{\max}; \\ c_1 * W_j + c_2 * (S_j - W_{\max}) & \text{if } W_j > W^g, \quad S_j > S_{\max}; \end{cases}$$

where: $W_{\max}$ is the maximum total working time allowed per day (9 hours);

W_j is the total working time of run j ;

W^g is the guaranteed minimum paid time (8 hours);

$S_{\max}$ is the maximum spread over time allowed per day (12 hours);

S_j is the spreadover time of run j;

c_1 is the pay rate of working hour ($25);

c_2 is the pay rate of spread over penalty ($20);

Objective Function:

$$\text{Min: } \sum_{j=1}^{n} c_j x_j \quad j = 1......n$$

$$\text{s.t. } \sum_{j=1}^{n} a_{ij} x_j = 1 \quad i = 1......m \qquad (1)$$

where c_j is the cost of run j

x_j is the jth run

$$x_j = \begin{cases} 1 & \text{if } j\text{th run is in the solution;} \\ 0 & \text{otherwise} \end{cases}$$

$$a_{ij} = \begin{cases} 1 & \text{if } i\text{th piece is in the } j\text{th run;} \\ 0 & \text{otherwise} \end{cases}$$

The inputs for this model are the pieces generated from the block cutting process. Each block is cut into different partitions while each partition consists of several pieces of work. And these pieces are generated while satisfying the constraint of a 2 hour minimum and 6 hour maximum piece length. These pieces are matched and combined into possible runs. When all the possible runs are generated by different piece matching, each possible run is checked for legibility by satisfying the following constraints: there is no overlap between the pieces; the break and change over occurs at the same relief point; total working time and spreadover time does not exceed maximum allowed time; each piece has a minimum and maximum length between 2 and 6 hours; and a meal break must exist after a maximum piece working time. With these legal runs generated through legibility check, a [0,1] matrix a_{ij} is generated indicating if piece i is in the jth run.

A test problem with two blocks is solved through this model using CPLEX. This test problem is composed of two given blocks. The block information is given in Table 2 and Table 3:

Table 2: Block 1 Data

Task number	Departing time	Arriving time	Relief point
1	706	718	TP
2	722	733	LP
3	742	811	SS
4	832	842	LP
5	849	913	SS
6	921	946	LP
7	949	1011	SS
8	1021	1046	LP
9	1049	1112	SS
10	1121	1146	LP
11	1149	1213	SS
12	1221	1248	LP
13	1251	1315	SS
14	1321	1348	LP
15	1351	1415	SS
16	1420	1448	LP
17	1451	1515	SS
18	1520	1548	LP
19	1551	1617	SS
20	1620	1650	LP
21	1653	1706	TP
22	1719	1742	TP
23	1806	1834	LP

Table 3: Block 2 Data

Task number	Departing time	Arriving time	Relief point
1	711	730	WH
2	738	752	FG
3	816	842	SS
4	851	916	LP
5	919	941	SS
6	951	1016	LP
7	1019	1041	SS
8	1051	1116	LP
9	1119	1143	SS
10	1151	1218	LP
11	1221	1245	SS
12	1251	1318	LP
13	1321	1345	SS
14	1351	1418	LP

15	1421	1445	SS
16	1453	1528	WH
17	1533	1608	SS
18	1622	1700	WH
19	1702	1740	SS
20	1742	1805	TP
21	1808	1822	LP
22	1833	1846	TP
23	1852	1906	LP

Given the block data above, this problem is solved as a set partitioning problem by the model above and the optimal solution is obtained using CPLEX. The results show that the optimal minimum cost is $626 with a total number of 3 runs generated.

The two sample blocks chosen only have few partition numbers, however, in practice there are too many variables to handle and the problem can not be solved in polynomial time. For example, for a middle sized problem with 100 vehicle blocks, each block usually has 10-30 different partitions with 3 to 4 pieces in each partition. At least 3000 pieces will be generated at this stage, as a result, millions of variables will be generated through different piece combinations into all possible runs for this problem. This problem is proven to be NP-hard (Fischetti et al. 1989) which means that solving this problem requires exponential time. As a result, a heuristic algorithm is needed to solve this NP-hard problem. In this thesis, a Modified Best-Fit-Decreasing Algorithm is proposed to solve the piece matching problem in the CSP which will be discussed later.

Overview of the solution process

The process of solving the crew scheduling problem is decomposed into two steps in this approach: block cutting and piece matching. In the block cutting process, all vehicle blocks are cut into pieces of work which generate an initial candidate set of pieces to be combined into runs in the second step. Of the large number of different partitions that could be cut for the same block, all possible partitions for each block are generated in this study with detailed piece information.

With the pieces generated from the first step, during the piece matching process, these pieces of work from the candidate set are matched and combined into feasible and legal runs. The matching criteria are both spatial and temporal. First, the matching pieces are geographically compatible which means the drivers are able to perform a changeover at the same relief point. Second, the starting time of the second piece should be no earlier than the ending time of the first piece, and last, all the time constraints related to labor rules need to be satisfied. The goal of the matching process is that all blocks are covered with a minimum total cost for runs. This approach formulates the matching part as a bin packing problem. The pieces are considered as items to be packed into bins, which are the runs. The algorithms used in the bin packing problem are studied and modified to apply to the crew scheduling problem. This two-step process is discussed in detail in the following sections.

The performance evaluation for the algorithm

It is usually difficult to evaluate the algorithm performance for a NP-hard problem. Originally, to evaluate the performance of a heuristic, the heuristic is programmed and tested on a representative sample size problem for empirical testing. However, for this bin packing problem, testing on a large problem set with known exact optimal solution is impossible. Two performance evaluation methods are used together to evaluate the results generated by the proposed approach.

(1) The estimated number of runs-from the TRB manual:

This evaluation method is based on the Transportation Research Board (TRB) manual-- Transit Cooperative Research Program (TCRP) Report 30: Transit scheduling: basic and advanced manuals. The most commonly used estimation technique is the total working time contained in the blocks divided by the working hours anticipated in each run. Usually, the target working hour is assumed to be 8 hours.

$$\text{Estimate number of runs} = \frac{\text{Total working time of all the blocks}}{\text{Target working hours per run}}.$$

In this study, the objective is to minimize the total cost of runs. To compare the results, it is assumed that these estimated runs have an equal cost of $200 ($25/pay rate*8/hour). However, this is an ideal situation that usually can not be achieved in practice.

The drawback of this estimation technique is that there are a number of split blocks with the small peak hour period pieces in the block set. If these pieces are matched together as split runs, the likelihood that they will form into runs with approximate 8 hour working time is questionable. Therefore, additional analysis of run possibilities is desirable.

(2) The average working time per run- LU criteria

The Lisbon Underground (LU) considers any schedule with an average of 4.5 driving hours per run is of very good quality (Cavique et al. 1999). There are mainly three different types of runs: straight run, split run, and tripper.

- A straight run is a continuous run with a short meal break in the middle. It starts from the beginning of the day to the end. It is considered the most cost efficient run.
- A split run is composed by two separate pieces with a long break in the middle. It is not desired but inevitable, since there are morning and afternoon peak hours which need more blocks to meet these demands.
- A tripper is a one piece run which is a short run with the most expensive cost. It is usually performed as an over time task.

Normally, the straight run has the longest working time and split run has a shorter working length. Trippers only last for few hours. This evaluation method is to get the average working time of runs (bin) for all these three types of run and it is calculated as:

$$\text{Average working hours per run} = \frac{\text{Total working hours of all runs}}{\text{Total number of runs}}.$$

Block Cutting Process

This section discusses the process of the block cutting which is the first step for solving the CSP. A solution method for cutting the blocks is presented.

Process introduction

In the block cutting process, the vehicle blocks are cut into pieces of work and an initial candidate set of feasible pieces of work is generated. A huge number of pieces will be generated since each block can have 10-30 different possible pieces of work and for all the blocks, there can be thousands of pieces of work which will lead to millions of possible runs.

Many partitions for each block are possible and it is very difficult to evaluate the partitions before forming runs. As a result, reducing the number of possible partitions of the blocks is the major task for this cutting process. Runs need to be cut intelligently. Some heuristic algorithms are proposed to reduce the number of variables: tabu search, genetic algorithm and column generation technique.

In this approach, the cutting strategy is mainly rule-based. Different combinations of the partitions will be generated for each block. If no constraints are placed on generating the partitions and each terminal could be a possible relief point, the number of possible partition of a block would be huge. So it is practical to have the blocks cut into pieces only by the defined relief point locations which are the service facilities that the drivers could get off the bus and have a meal break. Also, considering that the travel time and cost between different terminals for a changeover is not beneficial, in this study, the drivers are assumed to perform the changeover only at the same relief point.

There is usually a rule for defining the minimum and maximum length of a piece of work for practical reasons. In practice, a driver could not perform a piece of work for a long period without a break and driving while fatigued should be prohibited. It is also not beneficial to have a very short piece of work (like for 1 hour), since too many changeovers will cost time. In this study, the maximum and minimum piece length is set to be 6 hours and 2 hours, as generally used by bus companies.

All possible partitions for each block satisfying the rules are generated and this block cutting process is accomplished by a computer program. The constraints are: the block can be only cut at the relief point and each piece of work has a minimum and maximum length between 2 hours and 6 hours.

Solution Method

The block data given are composed by several tasks (trips), each task has the associated information of its departing time, arriving time, departing terminal and arriving terminal. Of all the terminals, several terminals are defined as the relief points where the rest services are provided.

A block may be cut at several points, resulting in different cutting patterns. The algorithm generates all possible partitions for each cut pattern (with the same total number of cuts), while satisfying the two following constraints:

1. Each piece should be cut only at a relief point;
2. Each piece of work has a time constraint between a minimum of 2 hours and a maximum of 6 hours.

The concept of mathematical induction is used in this cutting strategy. Mathematical induction is a mathematical method used to prove that a given statement is true of all natural numbers. It first proves that the first statement in the infinite sequence of statements is true, and then proves that if any one statement in the infinite sequence of statements is true, so is the next one. Once the first piece is cut, continue cutting the rest of the block in the same way as the first cut. This process is iterated until all possible partitions are generated for this block. In this run cutting process, for each block, the different piece partitions are generated by the following run cutting algorithm.

Run Cutting Algorithm:

Step 1: Given a whole block, first cut the smallest unit (which is composed by one or several tasks) satisfying the two constraints:

- Each unit (piece of work) can only be cut at a relief point;
- Each unit has a time constraint between 2 and 6 hours.

Step 2: Consider the remaining part of this block as a whole, continue cutting the smallest qualifying unit as in Step 1;

Step 3: Repeat Step 2 until the last piece of the block is cut, and then the first partition is generated. If the remaining part has a time length less than 2 hours, it is included in the preceding piece provided that all the constraints are satisfied.

Step 4: Fix the first piece cut from previous step, cut the second smallest qualifying unit from the remaining part of the block.

Step 5: Repeat Step 2 and 3 until last piece is cut, then the second partition is generated.

Step 6: Fix the first piece cut from previous step, cut the third smallest qualifying unit from the remaining part of the block.

Step 7: Repeat Step 2 and 3 until the last piece is cut, and then the third partition is generated.

Step 8: Repeat this iterative process until all possible partitions with the same first piece cut are generated.

Step 9: Take the whole block, cut the second smallest qualifying unit as the first piece, and repeat previous steps to generate new partitions.

Step 10: These steps iterates until the longest qualifying unit is cut as the first piece, then stop. And the cutting process completes with all the possible partitions generated.

A small example is provided for a detailed illustration: please refer to the block data from Table 2 for Block 1. The block information is shown in Figure 5. The numbered shade points indicate the relief points, where cuts can be made.

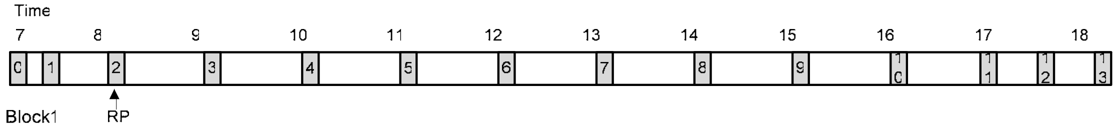


Figure 5: Block 1 information

Following the cutting algorithm, the first smallest qualifying unit to be cut is from origin 0 to relief point 3 - RP 3, then the second smallest cut for the next piece is at RP 6, repeat the steps and we can get the first partition as: RP3, RP6, RP9, and RP13 where the RP numbers indicate the cut points. Follow Step 4 in the algorithm, we can get the second partition as: RP3, RP6, RP10, RP13. And 6 other different partitions with the same first piece are generated following the procedure. Follow Step 9, we cut the second smallest piece as the first unit, the next partition is generated as: RP4, RP7, RP10, RP13. Repeat the steps until the last partition is generated as: RP6, RP10 and RP13.

When this cutting process is finished, all possible partitions satisfying the two constraints are generated and a large candidate set of the pieces are formed. A computer program is used for generating all possible partitions for the blocks. The concept of mathematical induction is applied in this program and Figure 6 shows a flowchart of the cutting strategy.

The task is defined to be a unit trip between two adjacent relief points. Label all the tasks q_{ij} for each block *k*. The problem for block cutting is essentially to list all possibilities of dividing the N ordered tasks into several subgroups.

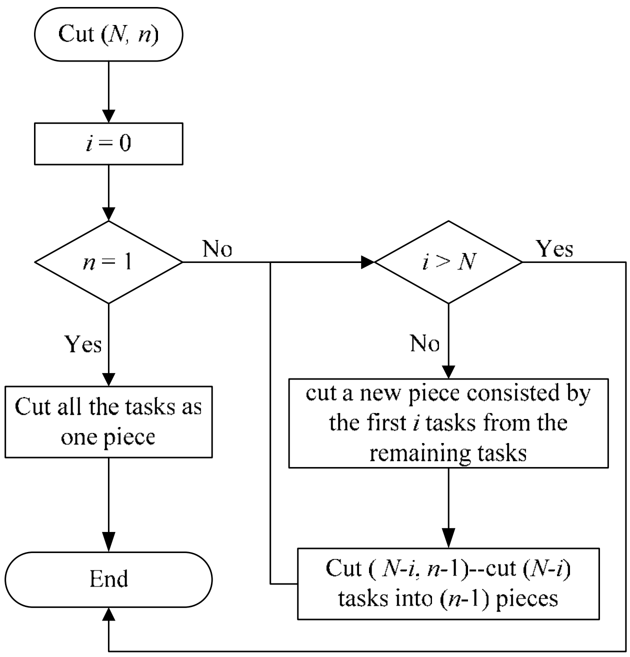


Figure 6 Block cutting strategy

An example of two of the partition results for one block are shown in Tables 4 and 5.

Table 4: Block Partition 1

5				6				6			
DP		SS		SS		SS		SS		SS	
0	316	449	456	449	456	813	816	813	816	1172	0

Table 5: Block Partition 2

6				5				6			
DP		SS		SS		SS		SS		SS	
0	316	513	516	513	516	813	816	813	816	1172	0

The numbers in the first row indicate where the cut are. For example: 5, 6, 6 means the first cut is at the 5th relief point and the second cut is at the next 6th relief point and the third cut is at the next 6th relief point in this block. The corresponding information for the relief point locations and piece departing and arriving time are also listed in the second and third row in the results for future uses in the piece matching process. A description of the results for one piece cut in Partition 1 is indicated in Table 6.

Table 6: Block Partition Codification

(6) cut location of relief point			
(SS) piece starting relief point		(SS) piece ending relief point	
(449) the ending time of preceding piece	(456) the starting time of next adjacent piece	(813) the ending time of preceding piece	(816) the starting time of next adjacent piece

An example to show how the time is calculated: 456= 7(hour)*60+36(minutes), so starting time 456 is in minutes and it means the starting time is 7:36. A complete list of all possible partitions for this block are listed in Appendix 1.

Piece Matching Process

This section discusses the process of the piece matching which is the second step for solving the CSP. A solution method for matching the pieces is presented.

Process introduction

The problem of generating runs from the pieces of work can be formulated as a matching problem (Edmonds, 1965). During this matching process, given the large set of all possible partitions of the blocks from the cutting process, the pieces of work are matched together to generate legal and feasible runs. In this matching process, all constraints that are imposed to ensure compliance with the working rules are considered.

In this approach, the piece matching problem is formulated as a bin packing problem. Each run is generated from scratch. The set of all the possible piece partitions are given and a Modified Best-Fit-Decreasing Heuristic is applied in this study for the run generation. The output is a set of bins with assigned pieces of work inside and an associated bin cost.

In this bin packing problem, assume each bin represents a run and the bins are filled with pieces of work. The capacity of the bin represents the time constraints for each run. Each bin has two types of capacity, one representing the total working time constraint, one representing the total spreadover time constraint. Both of the constraints need to be satisfied at the same time. A bin is called feasible and legal only if all the following constraints are satisfied:

1. The total piece working time does not exceed its working time capacity.
2. The total spreadover time does not exceed its spreadover capacity.
3. No overlap between the pieces exits and a piece following another one starts no earlier than its preceding piece.
4. The connecting point of the two pieces is the same relief point.
5. A meal break exists after a maximum piece working time of 6 hours.

All bins are generated when the blocks are completly covered. The work rules are described before.The bin cost is calculated as follows. Each driver is guaranteed a minimum of 8 hours pay per day, the working hours are no more than 9 hours per day and a spreadover penalty is paid to a driver at a premium rate of $20 for the time worked beyond 9 hours after the run begins.

The bin cost is assigned to each bin as below:

Cost = $25*Working hour + $20*(Spread over hour – 9)

- If a bin cost is less than $200 ($25*8), then assign the cost as $200. This is to guarantee the minimum paid time of 8 hours.

- If the spread over time is less than 9, then the Cost = $25*Working hour, which means there is no spreadover penalty.

Under different conditions of working time and spreadover time for each bin, the total cost is calculated as below:

$$C_P = \begin{cases} 25*8 & if\ W_P < 8, \quad S_P < 9; \\ 25*W_P & if\ W_P > 8, \quad S_P < 9; \\ 25*8 + 20*(S_P - 9) & if\ W_P < 8, \quad S_P > 9; \\ 25*W_P + 20*(S_P - 9) & if\ W_P > 8, \quad S_P > 9; \end{cases}$$

A set of bins representing the runs covering all pieces of work will be generated using the algorithm. A detailed description of the Modified Best-Fit-Decreasing Algorithm will be introduced in the following section.

Solution Method

A bin packing approach is proposed in this study for solving the piece matching problem as part of the solution to the crew scheduling problem. Based on the concept of the original bin packing problem, the matching problem of CSP with bin packing feature is defined as below:

Bin: A bin represents a run;

Item: An item is a piece of work cut from the block;

Capacity: The capacity constraints for the bin are the time constraints for the run with regards to the working rules. The capacity constraints are:

1) The maximum total working hour is 9 hours;
2) The maximum spreadover time is 12 hours;

Compatibility: This is to guarantee the two pieces to be matched are compatible. The criteria for checking the compatibility are:

1) There is no overlap between the two pieces, the second piece starts no earlier than the end of the first one.
2) The connecting relief points of the two pieces should be the same. This is to avoid performing a change over between different relief point locations so as to reduce the travel time and cost between relief points.

3) There should be a meal break after a maximum working period of 6 hours. And the minimum break is 30 minutes. It means if the total working time of the matched pieces exceeds 6 hours, there must be a break between the pieces.

The general idea of this matching part for the run generation is that each straight run usually starts with a morning piece. So, we can assume the first piece is fixed with the first morning piece of each block and filled in each bin. Then, pick from the remaining pieces to be filled in each bin with the constraints satisfied. Additional bins are opened when necessary. Finally, when all the tasks are covered and a bin set is generated, assign a cost to each bin.

In this construction process, one item is introduced at each step and a feasible solution is built in a greedy fashion. Of the set of the block partitions, for the first morning piece in each block, the longest partition of the earliest piece is chosen and filled in a bin as the first piece. The same idea is used for the rest of the piece selecting.

For the bin packing problem, as mentioned in Chapter 2, four efficient algorithms are commonly used that have the same order of time complexity, (Johnson et al. 1974). These are First-Fit Algorithm, Best-Fit Algorithm, First-Fit-Decreasing Algorithm and Best-Fit-Decreasing Algorithm. Among these, First-Fit-Decreasing and Best-Fit-Decreasing Algorithms are proved to be the most efficient based on the worst case performance study by Johnson et al. 1974.

In this thesis, the Best-Fit-Decreasing Algorithm is employed and modified such that it can be applied to the crew scheduling problem. A description of the Modified Best-Fit-Decreasing Algorithm for CSP is given below. The First-Fit-Decreasing Algorithm is also employed and modified in this study to be applied in the same problem context in a similar way as the Modified Best-Fit-Decreasing Algorithm. The results of the application of these two algorithms to the crew scheduling problem are compared in the case study.

Below is a list of the notations used in this algorithm.

K: set of blocks: $\{k_i\}$, $i = 1,2,...,m$;

Q: set of tasks: $\{q_{ij}\}$, $i = 1,2,...,m$; $j = 1,2,...,u_i$; q_{ij} is the *j*th task of block *i*;

R: set of partitions $\{R_{ij}\}$, $i = 1,2,...,m$, $j = 1,2,...,r_i$; R_{ij} is the *j*th partition of block *i*; r_i is the total number of partitions in block *i*;

P: set of pieces: $\{p_{ijk}\}$, $i = 1,2,...,m$, $j = 1,2,...,r_i$, $k = 1,2,...,v_{ij}$, p_{ijk} is the *k*th piece in partition j of block i, v_{ij} is the total number of pieces in partition *j*;

B: set of bins: $\{b_p\}$; $p = 1,2,...,n$; and *q* is the piece index in bin *p*, $q = 1,2,...,h$; denote *h* as the upper bound for the total piece number in a bin.

l_{ijk} : the length of piece p_{ijk};

$l_{\min}$: the minimum piece length required (2 hours);

$l_{\max}$: the maximum piece length allowed (6 hours);

$W_{\max}$: the maximum total working time allowed per day (9 hours);

W : set of the total working time of the bins $\{W_p\}$, $p = 1,2,...,n$;

W^g : the guaranteed minimum paid time (8 hours);

$S_{\max}$: the maximum spread over time allowed per day (12 hours);

S : set of the spreadover time of the bins $\{S_p\}$, $p = 1,2,...,n$;

$B^T_{\min}$: the minimum break time required (30 minutes);

T^s_{ijk} : the starting time of piece p_{ijk};

T^e_{ijk} : the ending time of piece p_{ijk};

L^s_{ijk} : the starting relief point of p_{ijk};

L^e_{ijk} : the ending relief point of p_{ijk};

C_p : the cost of bin *p*

c_1 : the pay rate of working hour ($25);

c_2 : the pay rate of spread over penalty ($20);

Modified Best-Fit-Decreasing Algorithm for generating bins:

Step 1: For each block in *K*, rank all possible partitions as *R*={ $R_{i1}, R_{i2}, ...R_{ir}$ }; select from *R* one piece with the earliest starting time and maximum working length. Label these pieces as selected;

Step 2: Assign each labeled piece from step 1 to an empty bin, and mark these bins as open;

Step 3: Sort all open bins generated from step 2 in a decreasing order based on their current total length, resulting a bin list *B*;

Step 4: Create a candidate piece list *L* consist by all non-labeled pieces with the same partition rank with the former pieces in the bins.

Step 5: Start with the lowest indexed bin from *B*, fill each bin with the next available piece selected from *L* based on the following constraints:

Select the piece of work p_{ijk} from *L* such that:

1. The piece has the same rank of partition with the former piece in the bin;

2. $\sum_{i=1}^{m}\sum_{j=1}^{r_i}\sum_{k=1}^{v_{ij}} T_{ijk}^{s} x_{i,j,k,p,q+1} - \sum_{i=1}^{m}\sum_{j=1}^{r_i}\sum_{k=1}^{v_{ij}} T_{ijk}^{e} x_{i,j,k,p,q} > 0, \forall p, \forall q < h-1$ (There is no time overlap between the pieces);

3. $\sum_{i=1}^{m}\sum_{j=1}^{r_i}\sum_{k=1}^{v_{ij}} L_{ijk}^{e} x_{i,j,k,p,q} - \sum_{i=1}^{m}\sum_{j=1}^{r_i}\sum_{k=1}^{v_{ij}} L_{ijk}^{s} x_{i,j,k,p,q+1} = 0, \forall p, \forall q < h-1$ (The connecting relief points must be the same between adjacent pieces);

4. $W_p < W_{\max}$ (The total working time does not exceed $W_{\max}$);

5. $S_p < S_{\max}$ (The total spread over time does not exceed $S_{\max}$);

6. If $W_p > l_{\max}$, select a piece p_{ijk} such that:

$\sum_{i=1}^{m}\sum_{j=1}^{r_i}\sum_{k=1}^{v_{ij}} T_{ijk}^{s} x_{i,j,k,p,q+1} - \sum_{i=1}^{m}\sum_{j=1}^{r_i}\sum_{k=1}^{v_{ij}} T_{ijk}^{e} x_{i,j,k,p,q} > B_{\min}^{T}; \forall p, \forall q < h-1$. (A run with a total working time longer than 6 hours must have a break longer than $B_{\min}^{T}$);

7. Of the selected feasible pieces satisfying the above 6 constraints, select p_{ijk} from L such that, T_{ijk}^{s} is the minimum one. (Select the piece with the earliest starting time to minimize the break time.)

8. If there is still a tie for those selected feasible pieces from constraint 7 (pieces are with the same starting time), select p_{ijk} from L such that, l_{ijk} is maximum. (Select the piece with longer working hour to pack the bin as full as possible.)

Step 6: Update the candidate list L by deleting the labeled pieces and the remaining pieces with a different partition rank number from the former pieces in the bins. And mark the bin as closed if b_p satisfies any of the following condition:

a. $W_p > W_{\max} - l_{\min}$, (the current total working time is more than 7 hours);

b. $S_p > S_{\max} - l_{\min}$, (the current spread over time is more than 10 hours);

c. No feasible piece p_{ijk} from L could be fit into b_p.

Step 7: Go back to step 5 and iterate until all the bins are marked as closed.

Step 8: Go back to step 1 until L is empty. Open a new bin when necessary.

Step 9: Assign each closed bin a bin cost according to the cost function:

$$C_p = \begin{cases} c_1 * W^g & \text{if } W_p < W^g, \quad S_p < S_{\max}; \\ c_1 * W_p & \text{if } W_p > W^g, \quad S_p < S_{\max}; \\ c_1 * W^g + c_2 * (S_p - W_{\max}) & \text{if } W_p < W^g, \quad S_p > S_{\max}; \\ c_1 * W_p + c_2 * (S_p - W_{\max}) & \text{if } W_p > W^g, \quad S_p > S_{\max}; \end{cases}$$

Step 10: Sum up the total number and cost of the bins generated and evaluate the results. The selecting process is mainly based on the feasibility and priority. The piece feasibility guarantees that this piece can be fit in the bin without violating the constraints and the piece priority guarantee that the best-fit piece is selected to get a better solution. A flow chart of this algorithm is drawn in Figure 7.

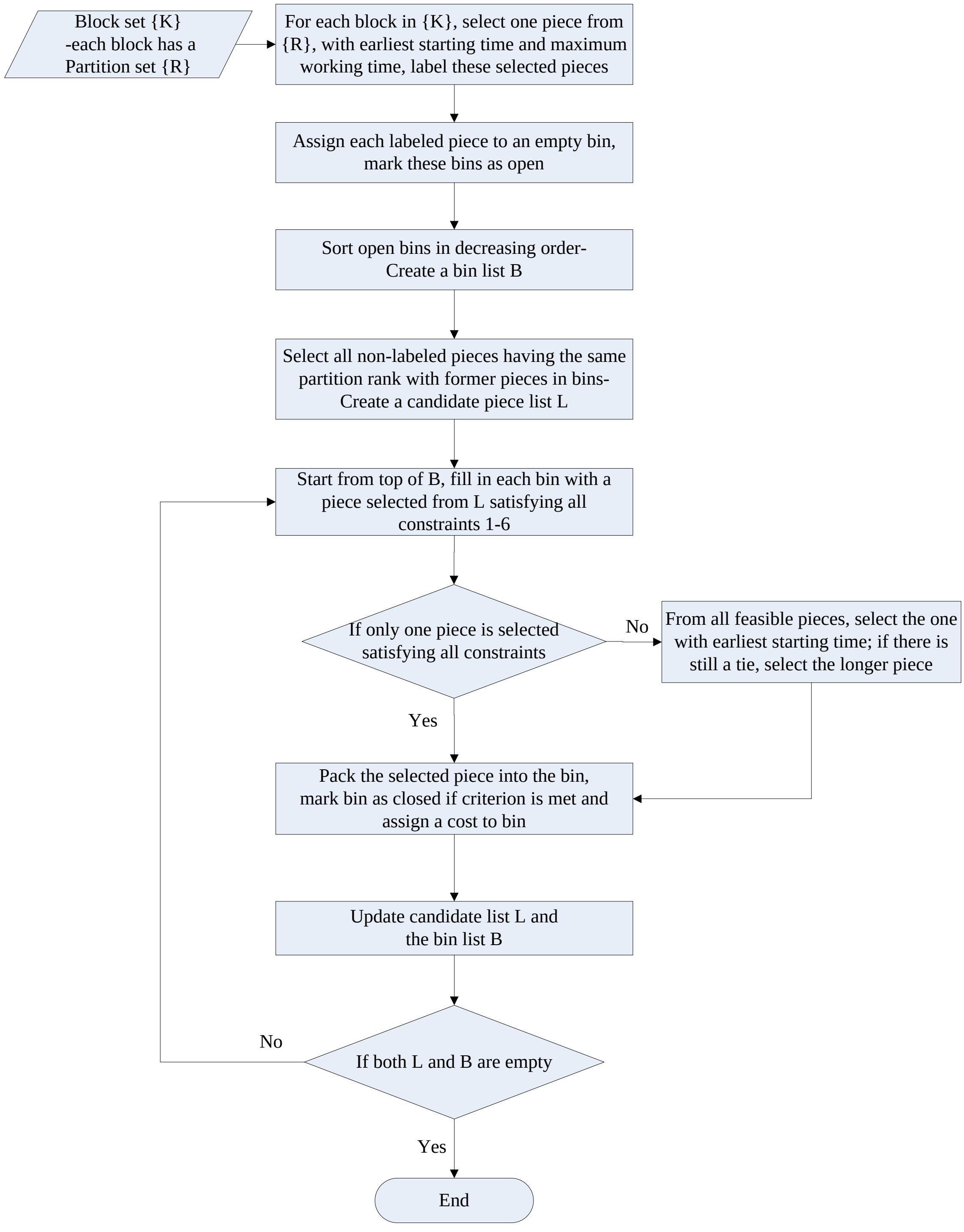


Figure 7 The flow chart of the modified Best-Fit-Decreasing Algorithm

To evaluate the results generated by this algorithm, three indexes are used for comparison. These evaluation indexes are mentioned in the previous sections. A summary of the results includes the following contents.

(1) The estimated total number of bins:

$$\text{Estimate number of bins} = \frac{\text{Total working time of all the blocks}}{\text{Target working hours per run (8 hours)}}$$

(2) The estimated total cost of bins:

$$\text{Estimate cost of bins} =$$
$$\text{hourly pay rate (\$25)} * \text{target working hour per run (8 hours)} * \text{Estimated number of bins}$$

(3) Average working hours:

$$\text{Average working hours per run} = \frac{\text{Total working hours of all runs}}{\text{Total number of runs}}$$

Test Problem Study

Three test problems are studied in this section to test the approach proposed above. Test problems 1 has two vehicle blocks, test problem 2 has three vehicle blocks and test problem 3 has 5 blocks which is the combination of the above two problems. The results are summarized at the end of this section and a detailed solution process and results are discussed for test problem 1 in this section.

Test problem 1 is the same problem used in Chapter 3, Section1. The optimal result is generated by CPLEX and will be used here for comparison with the heuristic results. Block data for two blocks are given: Block 9 and Block 10. Block information can be found in Section 1. First, use the block cutting algorithm to generate all possible partitions for each block. A detailed description description of the process is given in Chapter 3, Section 2. The partition results for these two blocks are attached in Appendices 1 and 2. The following description is to give a clear view of how the proposed Modified Best-Fit-Decreasing algorithm is used in practice to generate the runs.

1. In the set of all possible partitions for each block, select the partitions which have the earliest morning piece with the maximum piece length. See Tables 7 and 8 for the partition of Block 9 and 10.

Table 7: Block 9 Partition

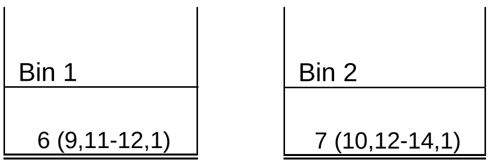
6	**3**	**4**
706-1213	1221-1515	1520-1834
LP-SS	SS-SS	SS-LP
6	**4**	**3**
706-1213	1221-1617	1620-1834
LP-SS	SS-SS	SS-LP

Table 8: Block 10 Partition

7	**3**	**6**
711-1245	1251-1528	1533-1906
LP-SS	SS-WH	WH-LP
7	**4**	**5**
711-1245	1251-1608	1622-1906
LP-SS	SS-SS	SS-LP
7	**5**	**4**
711-1245	1251-1700	1702-1906
LP-SS	SS-WH	WH-LP

2. Generate two empty bins for these two blocks, Bin 1 and Bin 2.

3. Fill in each bin with one of the first piece selected above, i.e. 6 and 7. Two bins are generated:



An example of the codification for one piece 6 (9,11,1) is described:

9	11-12	1
Block Number	Partition Rank	Piece Cut Position

4. Sort the current bins in a decreasing order according to its capacity. On top of the list is the bin with the longest piece, which means the bin is mostly filled with least residual space. So, the first bin is Bin 2 with a longer piece duration.

5. Generate a candidate list with non-labeled pieces.

3 (9,11,2)
4 (9,11,3)
4 (9,12,2)
3 (9,12,3)
3 (10,12,2)
6 (10,12,3)
4 (10,13,2)
5 (10,13,3)
5 (10,14,2)
4 (10,14,3)

6. Scan from the top of the candidate piece list. First, select the feasible pieces to fill into the bin according to the following constraints:

- There is no time overlap between the pieces;
- The connecting relief point location between these two pieces are the same;
- The total working time is no more than 9 hours;
- The total spreadover time is no more than 12 hours;
- Break time must exist after a long piece with more than 6 hours.
- Break time is at least 30 minutes.

For Bin 2, 3 feasible pieces are selected.

4 (9,11,3)
3 (9,12,3)
5 (10,13,3)

7. From the selected feasible pieces which satisfy all the constraints, based on the piece priority, select the piece with the earliest starting time. Piece 4 (9, 11, 3) is selected.
Bin 2 is:

4 (9,11,3)
7 (10,12-14,1)

8. Since each piece of work is cut with a minimum of 2 hours and maximum of 6 hours, a bin is closed if any of the following condition occurs:

- the current total working time is more than 7 hours;
- the current spread over time is more than 10 hours;
- No feasible piece could be filled into this bin.

Bin 2 is closed with a total working hour of 8 hours and 48 minutes and a total spreadover time of 11 hours and 23 minutes. The bin cost is assigned as:

Cost = $25*Working hour + $20*(Spread over hour – 9)

So, Bin 2 has a cost of $268.

9. Update the candidate list by deleting the labeled pieces and the non-labeled pieces with different partition rank. Since the selected piece is from the partition 6-3-4, the alternate partition 6-4-3 is deleted where piece 4 (9, 12, 2) and 3 (9, 12, 3) is deleted. And the already selected piece is also removed from the list. The updated candidate piece list is:

3 (9,11,2)
3 (10,12,2)
6 (10,12,3)
4 (10,13,2)
5 (10,13,3)
5 (10,14,2)
4 (10,14,3)

10. Select the next bin from the bin list: Bin 1 is the target bin. Repeat the steps 6, 7, 8 to fill the piece into the bin. There is a tie between 3 (10, 12, 2) and 4 (10, 13, 2). In this case select the longer piece. Bin 1 is generated as:

4 (10,13,2)
6 (9,11-12,1)

11. Bin 1 is closed with a total working hour of 8 hours and 24 minutes and a total spread over time of 9 hours and 2 minutes. The cost of Bin 1 is $211.

12. Update the candidate piece list as:

3 (9,11,2)
5 (10,13,3)

13. The current bin list is empty, open a new bin and select the piece with the earliest starting time from the candidate list. Filled in this piece into Bin 3.

14. Repeat the packing steps and Bin 3 is generated as:

5 (10,13,3)
3 (9,11,2)

15. Bin 3 is closed with a total working time of 5 hours and 38 minutes and a total spreadover time of 6 hours and 45 minutes. The cost of Bin 3 is $141. Since it is less than $200, the total cost is assigned as $200 to guarantee minimum pay.

16. Sum up the number and total cost of the bins generated.

3 bins are generated with a total cost of $677.

Bin 1	Bin 2	Bin 3
4 (10,13,2)	4 (9,11,3)	5 (10,13,3)
6 (9,11-12,1)	7 (10,12-14,1)	3 (9,11,2)

End

Test problem 1 summary:

Total cost of runs: $677

Total number of runs: 3 bins

The total spread over time of these 2 blocks is: 23.38 hours.

The total working time of these runs is: 22.83 hour

Estimated number of runs: 22.83hr / 8 =3 runs

Estimated total cost of runs: $600

Average working time for each run is: 22.83 / 3 = 7.6 hour

For test problem 2 and 3, a summary is given below to show the results:
Test problem 2 Summary:

Total cost of runs: $1281

Total number of runs: 6 runs

The total spread over time of these 3 blocks is 39.7 hours.

The total working time of these runs is 39.03 hours.

Estimated number of runs: 39.03 hr / 8hr = 5 runs

Estimated total cost of runs: $1000

Average working time for each run is: 39.03 / 6 = 6.51 hour

Test problem 3 Summary:

Total cost of runs: $1941

Total number of runs: 9 runs

The total spread over time of these 5 blocks is 63.08 hours.

The total working time of these runs is 62.72 hours.

Estimated number of runs: 62.72 hr / 8hr = 8 runs

Estimated total cost of runs: $1600

Average working time for each run is: 62.72 / 9 = 6.97 hour

The results for three test problems are summarized in Table 9 :

Table 9: Test Problems Results Summary

Test Problem	1	2	3
Total working time (hour)	22.83	39.03	62.72
Total spreadover time (hour)	23.38	39.7	63.08
Estimated number of runs	3	5	8
Estimated total cost	600	1000	1600
Total number of runs	3	6	9
Total cost	677	1281	1941
Average working time (hour)	7.8	6.51	6.97

These kinds of combinatorial problems usually include a large number of variables and finding an optimal solution for large size problem is usually very difficult. However, the optimal solution can be achieved for smaller size problems and these results can be used for comparing with the heuristic results generated. In this section, the optimal solutions for the above three test problems are obtained using CPLEX and are used to compare with the proposed heuristic results through BFD. A comparison of the optimal solution with the heuristic solution is given in Table 10 which displays the gap between the total cost of the optimal solution and the heuristic solution and the total number of runs generated. The gap is calculated using the following ration (HEU-OPT)/OPT where HEU and OPT represent the heuristic and the optimal solution costs respectively.

Table 10 Comparison between optimal solution and heuristic solution

	Test1 (2 blocks)			**Test2 (3 blocks)**			**Test3 (5 blocks)**		
	Optimal	**Heuristic**	**GAP**	**Optimal**	**Heuristic**	**GAP**	**Optimal**	**Heuristic**	**GAP**
Total cost ($)	626	677	0.075	1,200	1,281	0.063	1,673	1,941	0.138
Total num. runs	3	3	0.000	6	6	0.000	8	9	0.111

Chapter 4: Case Study-Solving the CSP for a Transit Bus Company

This chapter discusses a case study to solve a crew scheduling problem by using the approach proposed in this thesis. A crew scheduling problem for the RIDE ON bus system is solved here. The RIDE ON bus system is owned by the Montgomery County in Maryland and is coordinated by the Division of Transit Services. The Division plans, schedules and manages the County's own RIDE ON bus system. This bus system consists of 243 County owned and operated buses and 93 smaller contractor operated buses. The RIDE ON system provides over 22 million trips per year. It is designed to complement the service provided by other transit providers in the County including the Washington Metropolitan Area Transit Authority's Metrobus and Metrorail and the Maryland Mass Transit Administration's MARC commuter rail and MTA commuter bus systems. In this study we will solve the crew scheduling problem for RIDE ON based on the vehicle scheduling results they provided. A map indicating the service area of the RIDE ON bus system is shown in Figure 7 below:

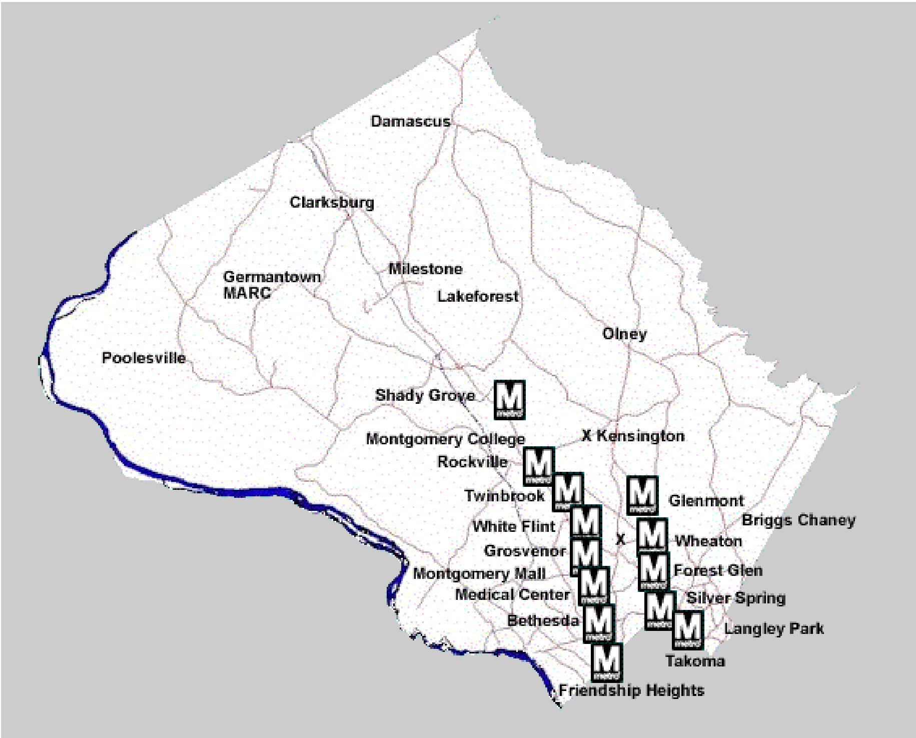


Figure 8 RIDE ON system map

Problem Objectives and Scope

The objective of this study is to solve a simplified crew scheduling problem with an output of crew runs and total costs. In this study, the vehicle block information is given, which is the output of the vehicle scheduling process. The objective is to find the minimum cost of runs while covering all the tasks. This process follows the two steps proposed in the approach. Two problems need to be solved: The first problem is how to cut the blocks into pieces of work with different partitions. The second problem is how to match and combine these pieces of work into feasible and legal runs with minimum costs, making sure that all tasks are covered.

The block information used in this case study is provided by a bus transit system in the State of Maryland. The information contains 73 block data which are part of the complete block data of the system. In this study, only the block information is given and the assumed working rules are mainly based on the TRB manual- TCRP report 30: Transit scheduling: basic and advanced manuals. The calculation for the cost of a run is assumed according to the general pay condition which is not set by the transit system. Above all, this study solved the CSP for the given 73 blocks with the assumed working rules and cost function.

Data Analysis

The bus transit system that has provided the block data serves one of the largest counties in the state of Maryland. This system provides over 22 million trips per year. The data provided by the bus transit system include: block information, relief points location and terminal codes. The database we have includes 73 blocks with a total number of 1299 tasks, 12 relief points and 89 terminals.

Relief Point Location

There are a total number of 89 terminals indicating each bus station, however, not every terminal can provide the services for a driver to change buses or take a break. Only 12 out of these 89 terminals are defined as relief point locations. The relief point location is a terminal where the meal and rest services are provided and the drivers can perform the changeover and take a meal break. A list of these relief points given by on the bus transit system is shown in the Table 11 below:

Table 11 Relief Point Location

Relief Point Locations	**Location Code**
German town Center	GN
Shady Grove Station	SE,SW
Twin Brook Station	TE,TW
Bethesda Station	BS
Medical Center	MD
Grosvenor	GR
Friendship Metro	FH
Silver Spring Station	SS
Takoma Station	TP
Forest Glen Station	FG
Wheaton Station	WH
Glenmont Station	GL

Block Information

The database of 73 vehicle blocks with 1299 tasks and the associate task information are given by on the bus transit system. Each task is provided with information of the starting and ending terminal of the trip, and the departing and arriving time. An example of parts of block 1 is shown in Table 12 below:

Table 12 Block 1 data

Task number	Block number	Departing terminal	Departing time	Arriving time	Arriving terminal
1	01	LP	516	525	TP
2	01	TP	532	541	LP
3	01	LP	546	555	TP

Transit service demand is at its highest level immediately before or after normal working hours due to the heavy commuting trips, and it varies greatly by time of the day. A large variation of vehicle demand occurs between the peak hours and off-peak hours where the ratio of demand for peak to off-peak is usually greater than 2. Accordingly, the length of the blocks varies greatly. In this project, the longest block starts as early as 4:30 in the morning and ends at 23:54 in the evening with a total working time of 19 hours and 24 minute. The shortest block is consisted by two separate morning and afternoon peak hour pieces with a total working time of only 4 hours and 11 minutes. This large variation in blocks is one of the most important features of transit scheduling and makes crew scheduling very complicated.

Of these 73 blocks, there are 32 straight blocks that start early in the morning and work continuously until the work is finished in the evening. The other 41 blocks are split blocks which are consisted by two separate pieces covering the demand of the morning peak hour and afternoon peak hours. Usually there is a long break of about 4-5 hours between the two periods. The split blocks take up 56% of the total blocks, this large percentage would usually cause the generation of split runs or trippers.
The total spreadover time of all the 73 blocks is 1000 hours and the total working time of the blocks is 712 hours. The average working time for each block is about 10 hours. A histogram for the total working time of each block is shown in the Figure 9:

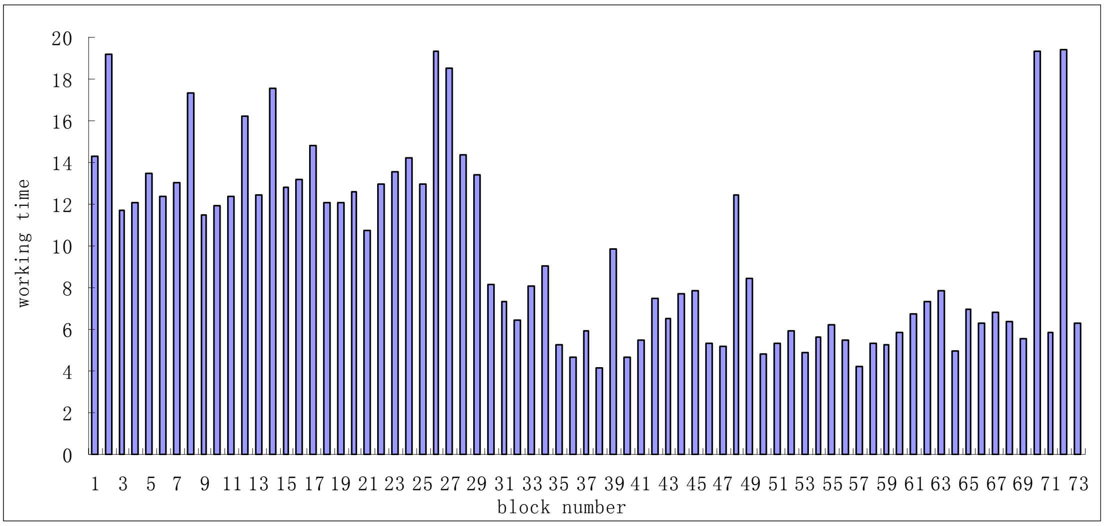


Figure 9 Block total working time

Assumptions

In this study, only the working rules for the crew scheduling is considered. Assumptions are made both on the time constraints and location constraints. The constraints made for block cutting are discussed in Chapter 3.

The calculation for the cost of a run is assumed according to the general pay condition. Each driver is guaranteed a minimum of 8 hours pay per day, the working hours are no more than 9 hours per day and a spread over penalty is paid to a driver at a premium rate of $20 for the time worked beyond 9 hours after the run begins. The spreadover penalty is an amount of pay granted to a driver for the time worked over a specified spreadover time. The total costs of the crew runs are the sum of the cost for each run generated.

Analysis of Results

In this section, the block cutting results and the piece matching results are analyzed and discussed. Different scenarios are provided for testing the algorithm and results are compared under each scenario.

Block cutting result

When the block cutting process is finished, all possible partitions satisfying the cutting constraints are generated for each block and a large candidate set of pieces are formed. An example of two partition results is shown in Tables 13 and 14:

Table 13: Block Partition result-partition 1

5				6				6			
DP		SS		SS		SS		SS		SS	
0	316	449	456	449	456	813	816	813	816	1172	0

Table 14: Block Partition result-partition 2

6				5				6			
DP		SS		SS		SS		SS		SS	
0	316	513	516	513	516	813	816	813	816	1172	0

The numbers in the first row indicate where the cut are, i.e. the cut position. For example: 5, 6, 6 means the first cut is at the 5^{th} relief point, the second cut is at the 6^{th} relief point and the third cut is at the 6^{th} relief point in this block. The corresponding information for the relief point locations and piece departing and arriving time are also listed in the second and third row in the results for future uses in the piece matching process. A description of the results for a one piece cut in Partition 1 is shown in Table 15 below:

Table 15: Block Partition result-partition codification

(6) cut location of relief point			
(SS) piece starting relief point		(SS) piece ending relief point	
(449) the ending time of former piece	(456) the starting time of this piece	(813) the ending time of this piece	(816) the starting time of next adjacent piece

Each of the 73 vehicle blocks are cut into several different partitions. Of these 73 blocks, a total of 2064 different partitions and thousands of pieces are generated. As mentioned before, these 73 blocks are consisted of 32 straight blocks and 41 split blocks. Of these 41 split blocks, each block has only one or two partitions satisfying the piece constraints. The partition number varies greatly between the different types of blocks. 35 split blocks have only one partition and for the straight blocks, the largest partition number can be up to 316 partitions. The complete partition results for several blocks are listed in the appendix. A histogram of the total partition number for each block is shown in Figure 10.

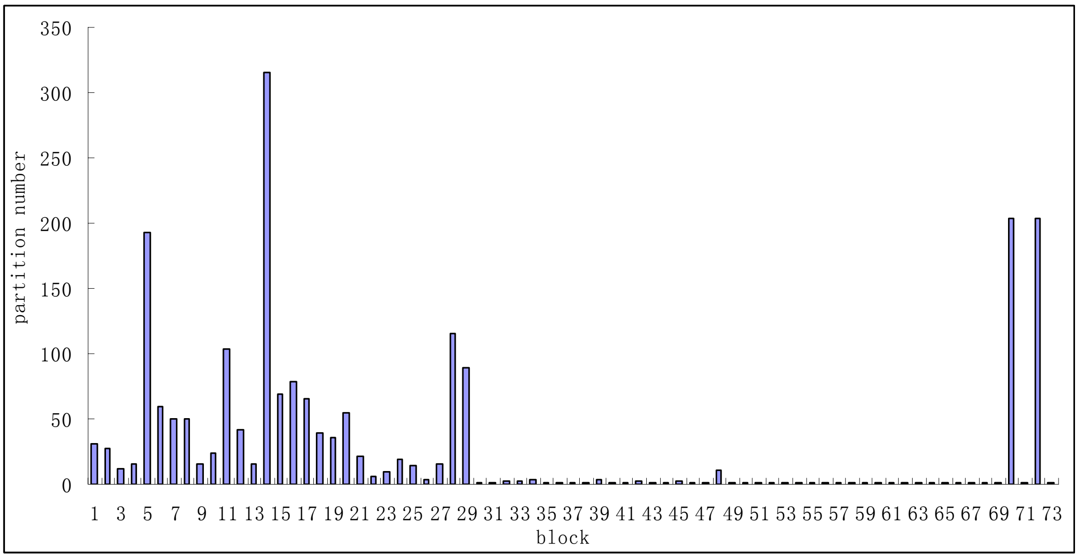


Figure 10 Block partition number

Piece matching result

In this matching process, the set of all possible piece partitions of the 73 blocks are generated and given from the block cutting process. The Modified Best-Fit-Decreasing Heuristic is applied here for the run generation. The output is a set of bins with assigned pieces of work inside and an associated bin cost. Two result files are generated for the run generation. One file is the bin list with pieces assigned to each bin. A complete result of the bin list is shown in Table 16.

Table 16 Bin results

Bin 1	Bin 15	Bin 30	Bin 46	Bin 65	Bin 81	Bin 101	Bin 120
1 6 1	15 20 1	30 1 1	46 1 1	65 1 1	6 30 3	50 1 2	71 1 2
2 17 2	17 61 3	Bin 31	42 2 2	Bin 66	Bin 82	Bin 102	Bin 121
Bin 2	Bin 16	31 1 1	Bin 47	66 1 1	43 1 2	46 1 2	20 21 3
2 17 1	16 22 1	Bin 32	47 1 1	Bin 67	Bin 83	Bin 103	Bin 122
9 11 2	21 18 2	32 1 1	22 5 3	67 1 1	15 20 3	65 1 2	5 192 4
Bin 3	Bin 17	14 277 2	Bin 48	70 192 2	Bin 84	Bin 104	Bin 123
3 1 1	17 8 1	37 1 2	48 6 1	Bin 68	63 1 2	38 1 2	61 1 2
28 113 3	19 23 2	Bin 33	Bin 49	68 1 1	Bin 85	42 2 3	Bin 124
Bin 4	Bin 18	33 1 1	49 1 1	22 5 2	25 9 3	Bin 105	24 19 4
4 11 1	18 19 1	1 31 2	Bin 50	Bin 69	Bin 86	60 1 2	Bin 125
29 88 2	39 2 2	Bin 34	50 1 1	69 1 1	62 1 2	Bin 106	54 1 2
Bin 5	Bin 19	34 1 1	Bin 51	28 113 2	Bin 87	58 1 2	Bin 126
5 36 1	19 23 1	5 192 2	51 1 1	4 11 3	44 1 2	27 9 4	1 31 4
6 30 2	15 20 2	Bin 35	68 1 2	Bin 70	Bin 88	Bin 107	Bin 127
Bin 6	Bin 20	35 1 1	Bin 52	70 159 1	9 11 3	73 1 2	32 2 3
6 30 1	20 18 1	7 22 2	52 1 1	25 9 2	Bin 89	70 195 4	Bin 128
32 2 2	16 78 3	Bin 36	Bin 53	Bin 71	23 4 3	Bin 108	40 1 2
Bin 7	Bin 21	36 1 1	53 1 1	71 1 1	Bin 90	66 1 2	Bin 129
7 20 1	21 2 1	Bin 37	Bin 54	72 192 2	26 2 3	Bin 109	7 22 3
1 31 3	19 23 3	37 1 1	54 1 1	Bin 72	Bin 91	35 1 2	Bin 130
Bin 8	Bin 22	11 48 2	Bin 55	72 159 1	67 1 2	Bin 110	17 61 4
8 47 1	22 5 1	Bin 38	55 1 1	27 9 2	Bin 92	47 1 2	Bin 131
33 2 2	23 4 2	38 1 1	17 61 2	Bin 73	48 6 3	Bin 111	59 1 2
Bin 9	Bin 23	56 1 2	Bin 56	73 1 1	Bin 93	45 2 2	Bin 132
9 11 1	23 4 1	Bin 39	56 1 1	26 2 2	11 48 3	2 17 4	16 78 4
10 13 2	48 6 2	39 1 1	Bin 57	Bin 74	45 2 3	Bin 112	Bin 133
Bin 10	Bin 24	18 22 2	57 1 1	12 33 2	Bin 94	18 22 3	8 50 3
10 12 1	24 6 1	Bin 40	Bin 58	21 18 3	31 1 2	Bin 113	Bin 134
5 192 3	Bin 25	40 1 1	58 1 1	Bin 75	Bin 95	51 1 2	33 2 3
Bin 11	25 9 1	Bin 41	24 19 2	16 78 2	34 4 2	Bin 114	Bin 135
11 45 1	24 19 3	41 1 1	Bin 59	55 1 2	Bin 96	10 13 3	28 113 4
14 277 3	Bin 26	Bin 42	59 1 1	Bin 76	3 11 3	Bin 115	Bin 136
Bin 12	26 2 1	42 1 1	Bin 60	3 1 2	Bin 97	69 1 2	14 277 4
12 2 1	Bin 27	57 1 2	60 1 1	Bin 77	70 198 3	8 50 4	Bin 137
13 11 2	27 9 1	Bin 43	Bin 61	27 9 3	Bin 98	Bin 116	12 33 4
Bin 13	Bin 28	43 1 1	61 1 1	72 197 4	72 197 3	36 1 2	Bin 138
13 11 1	28 11 1	8 50 2	Bin 62	Bin 78	26 2 4	Bin 117	39 2 3
12 33 3	4 11 2	Bin 44	62 1 1	2 18 3	Bin 99	41 1 2	Bin 139
Bin 14	Bin 29	44 1 1	Bin 63	Bin 79	52 1 2	Bin 118	29 88 4
14 1 1	29 19 1	Bin 45	63 1 1	30 1 2	Bin 100	53 1 2	Bin 140
	29 88 3	45 1 1	Bin 64	Bin 80	13 11 3	Bin 119	34 1 3
		20 21 2	64 1 1	49 1 2		64 1 2	

An example of the codification for each piece in the bin is shown in Table 17:

Table 17 Bin result codification

1	6	1
Block Number	Partition Rank	Piece Cut Position

The other result file is a summary of the bin results including a cost list for each bin with a total working time, and a summary listing: total bin number, total spreadover time, total bin cost, total working time, estimated bin number, estimated bin cost and the average working time. Complete results of the bin cost are shown in Table 18.

Table 18 Bin cost results

cost bin 1: 205	cost bin 19: 210	cost bin 37: 200	cost bin 55: 200
work time:8.23	work time:8.43	work time:6.5	work time:5.12
cost bin 2: 210	cost bin 20: 245	cost bin 38: 200	cost bin 56: 200
work time:8.12	work time:8.23	work time:5.38	work time:2.1
cost bin 3: 256	cost bin 21: 262	cost bin 39: 218	cost bin 57: 200
work time:8.93	work time:8.97	work time:7.5	work time:2.15
cost bin 4: 204	cost bin 22: 226	cost bin 40: 200	cost bin 58: 200
work time:8.17	work time:8.78	work time:2.67	work time:5.3
cost bin 5: 215	cost bin 23: 231	cost bin 41: 200	cost bin 59: 200
work time:8.32	work time:8.9	work time:2.47	work time:3.27
cost bin 6: 228	cost bin 24: 200	cost bin 42: 200	cost bin 60: 200
work time:8.2	work time:5.17	work time:4.33	work time:2.43
cost bin 7: 239	cost bin 25: 244	cost bin 43: 233	cost bin 61: 200
work time:8.85	work time:8.75	work time:7.85	work time:4.33
cost bin 8: 240	cost bin 26: 200	cost bin 44: 200	cost bin 62: 200
work time:8.35	work time:5.83	work time:3.15	work time:3.43
cost bin 9: 210	cost bin 27: 200	cost bin 45: 204	cost bin 63: 200
work time:8.4	work time:5.5	work time:7.4	work time:3.42
cost bin 10: 216	cost bin 28: 201	cost bin 46: 200	cost bin 64: 200
work time:8.32	work time:8.05	work time:4.4	work time:2.47
cost bin 11: 273	cost bin 29: 246	cost bin 47: 200	cost bin 65: 200
work time:8.68	work time:8.27	work time:6.07	work time:3.47
cost bin 12: 219	cost bin 30: 200	cost bin 48: 200	cost bin 66: 200
work time:8.57	work time:3.65	work time:5.83	work time:3.45
cost bin 13: 261	cost bin 31: 200	cost bin 49: 200	cost bin 67: 242
work time:8.62	work time:3.18	work time:3.68	work time:8.77
cost bin 14: 200	cost bin 32: 256	cost bin 50: 200	cost bin 68: 200

work time:5.73	work time:8.28	work time:2.25	work time:6.13
cost bin 15: 246	cost bin 33: 200	cost bin 51: 205	cost bin 69: 254
work time:8.68	work time:5.77	work time:5.8	work time:7.85
cost bin 16: 217	cost bin 34: 200	cost bin 52: 200	cost bin 70: 252
work time:8.42	work time:4.95	work time:2.97	work time:8.77
cost bin 17: 200	cost bin 35: 200	cost bin 53: 200	cost bin 71: 244
work time:7.9	work time:6.43	work time:2	work time:8.83
cost bin 18: 262	cost bin 36: 200	cost bin 54: 200	cost bin 72: 224
work time:8.32	work time:2	work time:3.62	work time:8.73
cost bin 73: 216	cost bin 91: 200	cost bin 109: 200	cost bin 127: 200
work time:8.23	work time:3.42	work time:3.15	work time:2.55
cost bin 74: 200	cost bin 92: 200	cost bin 110: 200	cost bin 128: 200
work time:5.07	work time:3.37	work time:3.2	work time:2
cost bin 75: 200	cost bin 93: 200	cost bin 111: 200	cost bin 129: 200
work time:5.17	work time:4.95	work time:7.87	work time:2.98
cost bin 76: 200	cost bin 94: 200	cost bin 112: 200	cost bin 130: 200
work time:5.82	work time:4.17	work time:2.1	work time:3.73
cost bin 77: 236	cost bin 95: 200	cost bin 113: 200	cost bin 131: 200
work time:8.73	work time:4.23	work time:2.68	work time:2
cost bin 78: 200	cost bin 96: 200	cost bin 114: 200	cost bin 132: 200
work time:5.88	work time:2.82	work time:2.73	work time:2.22
cost bin 79: 200	cost bin 97: 200	cost bin 115: 200	cost bin 133: 200
work time:4.52	work time:5.35	work time:7.05	work time:2.5
cost bin 80: 200	cost bin 98: 200	cost bin 116: 200	cost bin 134: 200
work time:4.73	work time:7.27	work time:2.7	work time:2.28
cost bin 81: 200	cost bin 99: 200	cost bin 117: 200	cost bin 135: 200
work time:3.9	work time:2.95	work time:2.98	work time:2.87
cost bin 82: 200	cost bin 100: 200	cost bin 118: 200	cost bin 136: 200
work time:3.82	work time:2.92	work time:2.88	work time:5.67
cost bin 83: 200	cost bin 101: 200	cost bin 119: 200	cost bin 137: 200
work time:4.22	work time:2.55	work time:2.47	work time:4.18
cost bin 84: 200	cost bin 102: 200	cost bin 120: 200	cost bin 138: 200
work time:4.43	work time:3.32	work time:2.4	work time:3.57
cost bin 85: 200	cost bin 103: 200	cost bin 121: 200	cost bin 139: 200
work time:4.05	work time:3.5	work time:2	work time:2.07
cost bin 86: 200	cost bin 104: 200	cost bin 122: 200	cost bin 140: 200
work time:3.93	work time:4.93	work time:2.3	work time:3.85
cost bin 87: 200	cost bin 105: 200	cost bin 123: 200	
work time:4.57	work time:3.42	work time:2.42	
cost bin 88: 200	cost bin 106: 200	cost bin 124: 200	

work time:3.23	work time:6.9	work time:2.97	
cost bin 89: 200	cost bin 107: 200	cost bin 125: 200	
work time:4.18	work time:6.75	work time:2	
cost bin 90: 200	cost bin 108: 200	cost bin 126: 200	
work time:5.83	work time:2.83	work time:2.67	
Total bin cost:$29,150			
Total bin number:140			
Total spreadover time:1000.1 hour			
Total work time:706.82 hour			
Estimated bin number:89			
Estimated bin cost: $17,800			
Average work time:5.05 hour			

A summary of the run results for the 73 blocks are listed in Table 19 below:

Table 19 Bin results summary

Total bin number: 140
Total bin cost: $ 29,150
Total spreadover time:1000 hour
Total working time:707 hour
Estimated bin number: 89
Estimated bin cost: $17,800
Average work time: 5.05 hour

The estimated number of bins and the estimated total cost usually can not be reached in practice. Of these 73 blocks, there are 32 straight blocks and the other 41 blocks are split blocks consisted by two separate pieces covering the morning peak hour and afternoon peak hour. Since the split blocks take up to 56% of the total blocks, this large percentage would usually cause the generation of split runs or trippers which are not efficient runs. At the same time, the estimated bin cost is an ideal situation without considering the spreadover time penalty.

The current solution of the problem and total cost is not communicated with the company, and it is usually difficult to compare the results with other people's work directly. The average working time is chosen as a measure of effectiveness. The Lisbon Underground (LU) considers then any schedule with an average of 4.5 driving hours per run is of very good quality (Cavique et al. 1999). Compared to this, the resulting average working time of 5.05 hours is good.

The optimal solution can be obtained using CPLEX for this problem. The result shows that the gap between optimal and heuristic solution is 0.1135 for the total cost and 0.0786 for the total number of runs generated. However, tests results show that when the problem size grows, CPLEX is not able to find the solution for the larger size problems. For example, when the block size doubles, the total number of possible runs grows almost 5 times compared to the original problem and CPLEX could not solve the problem at this size. Test results show that CPLEX stops working when the total number of blocks reaches 89. The test results are shown in Table 20.

Table 20: Tests results for different sizes

Block number	Method	Total task	Possible num. of runs	Total cost ($)	Total num. of runs	Solution time(sec)
73 blocks	OPT	1299	725,328	25,842.09	129	32.17
	BFD		-	29,150.00	140	0.13
	GAP		-	0.1135	0.0786	-
75 blocks	OPT	1361	781,184	25,843.58	129	23.16
	BFD		-	29,880.00	143	0.13
	GAP		-	0.1351	0.0979	-
83 blocks	OPT	1548	1,388,373	25,842.83	129	52.92
	BFD		-	33,251.00	159	0.14
	GAP		-	0.2228	0.1887	-
86 blocks	OPT	1637	1,656,666	25,844.50	129	39.17
	BFD		-	34,719.00	166	0.16
	GAP		-	0.256	0.223	-
88 blocks	OPT	1693	2,340,026	25,842.83	129	51.34
	BFD		-	35,614.00	170	0.16
	GAP		-	0.2744	0.2412	-
89 blocks	OPT	1714	2,423,180	N/A	N/A	N/A
	BFD		-	36,061.00	172	0.36
	GAP		-	-	-	-
90 blocks	OPT	1737	2,512,768	N/A	N/A	N/A
	BFD		-	36,282.00	173	0.36
	GAP		-	-	-	-
146 blocks	OPT	2598	3,558,648	N/A	N/A	N/A
	BFD		-	58,299.00	280	0.30
	GAP		-	-	-	-

Result Improvements

In this section, several tests are conducted under different scenarios to improve the current run results.

First, the Modified First-Fit-Decreasing Algorithm (MFFD) is applied to the blocks to see if there is any improvement for the bin results. This Modified First-Fit Algorithm adopted the same idea with the Modified Best-Fit-Decreasing Algorithm only except that the bins are not sorted beforehand in a decreasing order based on their current total length.

Second, the Modified Best-Fit (MBF) and Modified First-Fit algorithms (MFF) are applied to the blocks which adopted the same idea with the MBFD and MFFD except that the first piece selected for each bin is not the longest piece in the partition set. Instead, the second longest piece is selected. After testing the second longest piece, the third longest piece is selected and so on.

Third, different types of blocks are grouped and tested. The Modified Best-Fit-Decreasing Algorithm (MBFD) is applied separately both to the straight blocks group and the split blocks group. This is done to test the performance of this algorithm on different types of vehicle blocks.
The results for each scenario are summarized in Table 21.

Table 21 Result summary under different scenarios

Algorithm	Type of Block	Total bin cost ($)	Estimate bin cost ($)	Total bin number	Estimate bin number	Average working time(hr)	Total working time(hr)	Total spreadover (hr)
MBFD	All blocks (73 blocks)	$29,150	$17,800	140	89	5.05	706.82	1000.1
	Straight blocks (32 blocks)	$15,289	$11,400	72	57	6.27	451.18	454.07
	Split blocks (41 blocks)	$15,605	$6,600	78	33	3.36	261.82	546.03

MFFD	All blocks	$29,117	$17,600	140	88	5.03	703.78	1000.1
	Straight blocks	$14,899	$11,200	70	56	6.38	446.82	454.07
	Split blocks	$15,631	$6,600	78	33	3.34	260.22	546.03
MBF-2nd longest piece	All blocks	**$28,465**	**$17,600**	**136**	**88**	**5.16**	**701.88**	**1000.1**
	Straight blocks	$14,712	$11,400	68	57	6.59	448.25	454.07
	Split blocks	**$15,605**	**$6,600**	**78**	**33**	**3.36**	**261.82**	**546.03**
MFF-2nd longest piece	All blocks	$28,934	$17,600	140	88	5	700.6	1000.1
	Straight blocks	**$14,508**	**$11,400**	**68**	**57**	**6.59**	**448.3**	**454.07**
	Split blocks	$15,631	$6,600	78	33	3.34	260.22	546.03
MBF-3rd longest piece	All blocks	$28,996	$17,800	138	89	5.1	704.3	1000.1
MFF-3rd longest piece	All blocks	$28,902	$17,600	138	88	5.07	699.85	1000.1
MBF-4th longest piece	All blocks	$28,533	$17,600	136	88	5.15	700.72	1000.1
MFF-4th longest piece	All blocks	$29,364	$17,800	143	89	4.94	706.57	1000.1
MBF-5th longest piece	All blocks	$28,790	$17,600	137	88	5.13	703.08	1000.1
MFF-5th longest piece	All blocks	$29,170	$17,600	140	88	5.02	702.47	1000.1
MBF-shortest piece	All blocks	$28,768	$17,600	139	88	5.02	697.37	1000.1
MFF-shortest piece	All blocks	$29,228	$17,600	142	88	4.92	698.75	1000.1

First, the results show that there is no major difference between using the MBFD, MBF, MFF and MFFD. The best results for different scenarios are shown in bold in Table 18. The best results of the run set for 73 blocks occurs under the scenario of selecting the second longest piece with the Modified Best-Fit Algorithm. This result generates a total number of 136 bins with an average working time of 5.16 hours and total cost of $28,465.

For the straight blocks, the best result occurs under the scenario of selecting the second longest piece with the Modified First-Fit Algorithm. This result generates a total number of 68 bins with an average working time of 6.59 hours and total cost of $14,508. For the split blocks, the best results are generated for either applying the MBFD or MBF with the second longest piece selected. This result generates a total number of 78 bins with an average working time of 3.36 hours and total cost of $15,605.

The results under different scenarios are discussed here. First, a comparison of the total cost of bins between different algorithms is shown in Figure 11. It shows that the Modified Best-Fit Algorithm of selecting the second longest piece generates the lowest cost of $28,465. A total of 136 bins are generated. The runs have an average working time of 5.16 hours which is nearly 15% higher than the 4.5 hours LU considered as good quality. Though the gap between the total cost and the estimated total cost is large, it should be mentioned that the estimated cost is in an ideal situation that is hard to achieve in practice.

Second, a comparison of the total cost between different algorithms for only straight blocks is shown in Figure 12. It shows that for the straight blocks, the Modified First-Fit Algorithm with selecting the second longest piece generates the lowest cost of $14,508. A total number of 68 bins are generated and the runs have an average working time of 6.59 hours which is 46% higher than the 4.5 hours LU considered as good quality. The gap between the total cost and the estimated total cost is 27.3%, and the gap between the total bin number and estimated bin number is 19.3%.

Third, a comparison of the total cost between different algorithms for only split blocks is shown in Figure 13. It shows that for the split blocks, the Modified Best-Fit Algorithm with selecting the second longest piece generates the lowest cost of $15,605. A total of 78 bins are generated and the runs have an average working time of 3.36 hours which is 25% lower than the 4.5 hours. The split blocks are the major reason for the high cost of the runs and the large number of runs needed. Since these split blocks are covering the morning and afternoon peak hours, they are difficult to combine into a straight run. Split runs and trippers are generated for this type of blocks. Though split runs and tripper are not desired, they are inevitable in practice. Also, in this case study, the high percentage of 56% split blocks is a reason for the high total cost.

These results indicate that the type of the runs, and the percentage of different types of runs of the total blocks have a significant influence on the results. Figure 14 indicates the gaps between total cost and estimated total cost of runs, as well as the total number of runs and the estimated total number of runs for different types of block pools: all blocks, only straight blocks and only split blocks. It is clear that the straight blocks have a smaller gap with the estimated results than the split blocks.

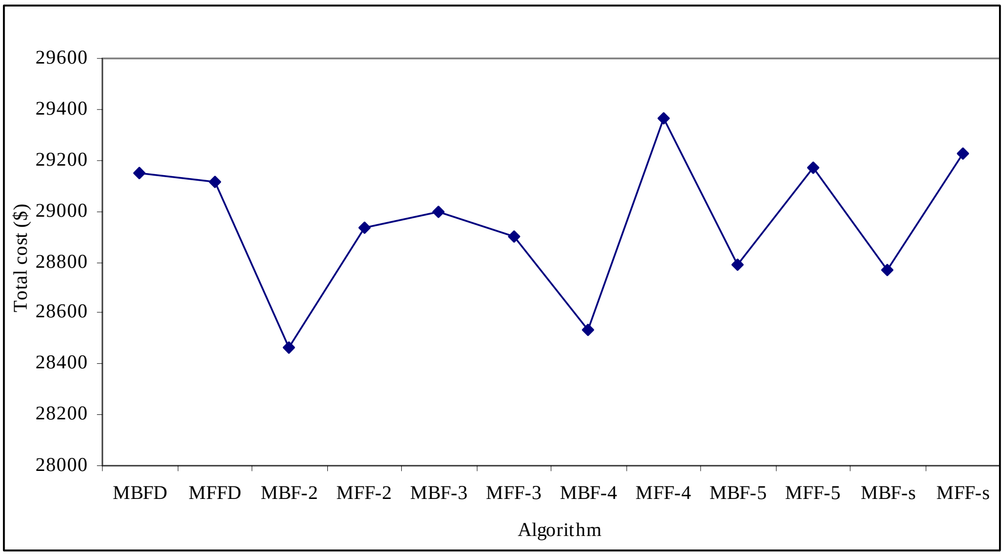


Figure 11 Comparison of total cost between different algorithms

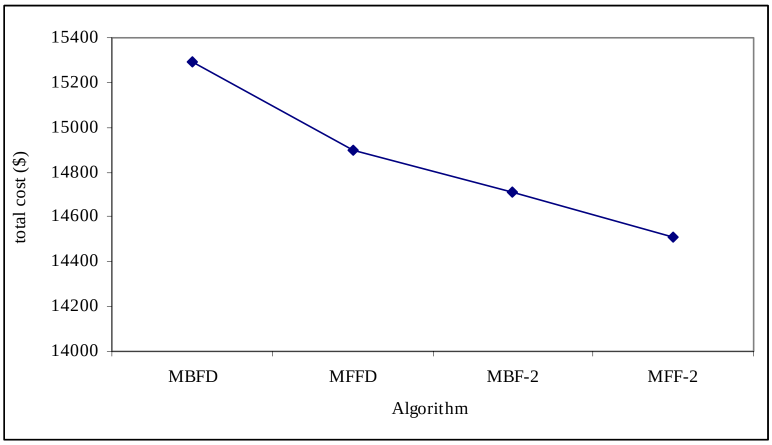


Figure 12 Comparison of total cost between different algorithms for straight blocks

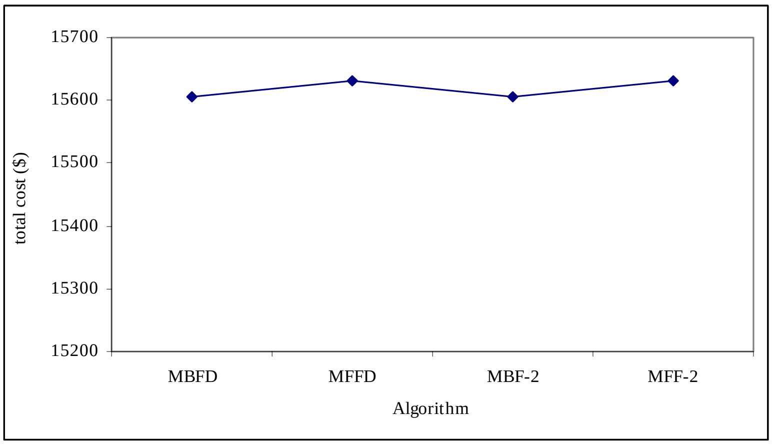


Figure 13 Comparison of total cost between different algorithms for split blocks

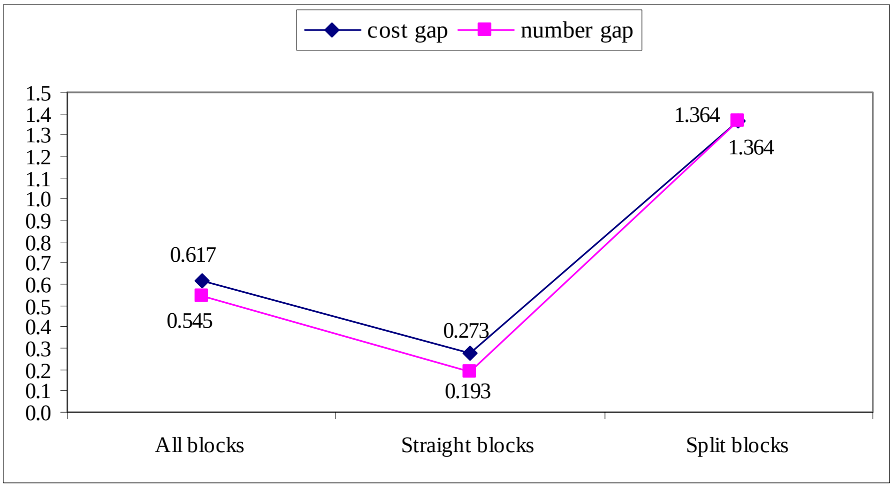


Figure 14 Comparison of total cost gap and total number gap between different blocks

Considering the drawback of the run estimation technique, namely that there are a number of split blocks with small peak hour period pieces in the block set and these pieces are matched together as split runs, the likelihood that they will form runs with approximate 8 hour working time is questionable. Therefore, additional performance evaluation method is desirable. The LU considers 4.5 hours average working time per run for a schedule is of very good quality, and this value is used to compare with the average working time generated in this study under different scenarios. In Figure 15, the average working times obtained with different algorithms are compared with the LU criteria. This average working time results from all the blocks. Figure 16 is the comparison for only the straight blocks, which indicated a higher average working time. Figure 17 compares the split blocks only. The average working time is shorter for this case and the results obtained from different algorithms show little difference. This indicates that the split blocks are causing a shorter average working time for the runs generated.

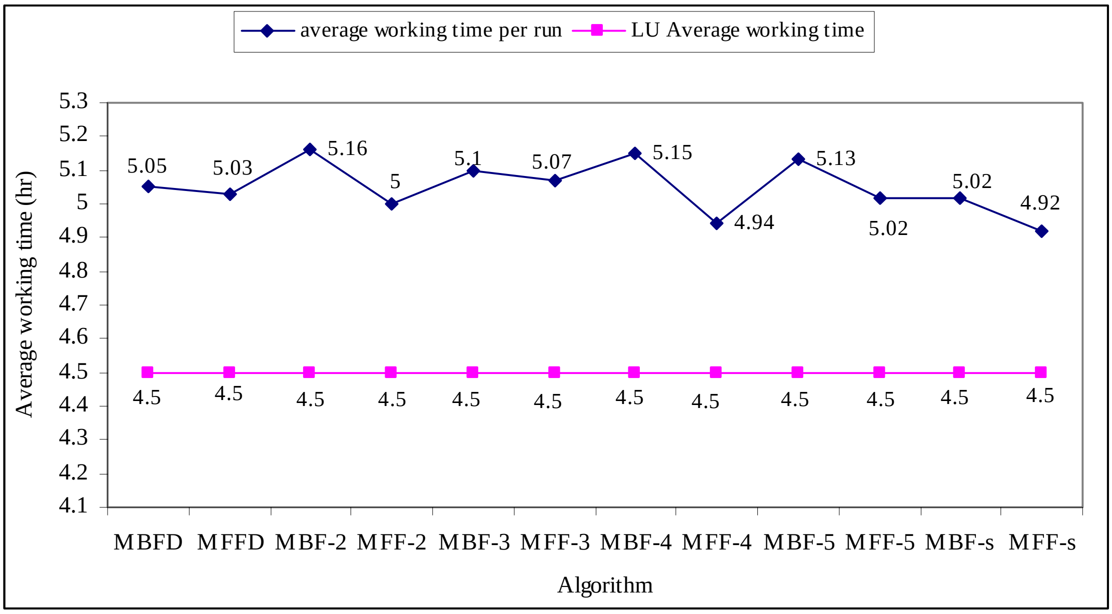


Figure 15 Comparison of average working time with LU criteria for different algorithms

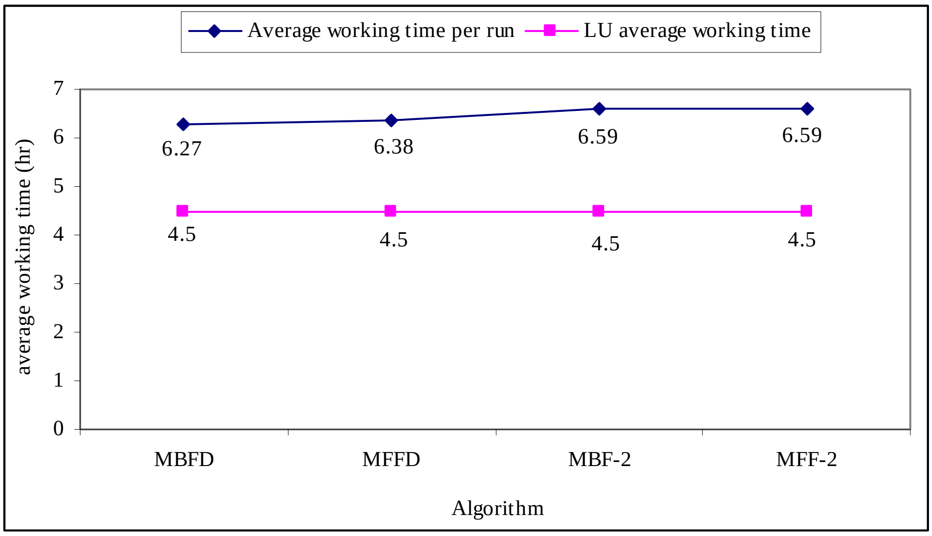


Figure 16 Comparison of average working time with LU criteria between algorithms for straight blocks

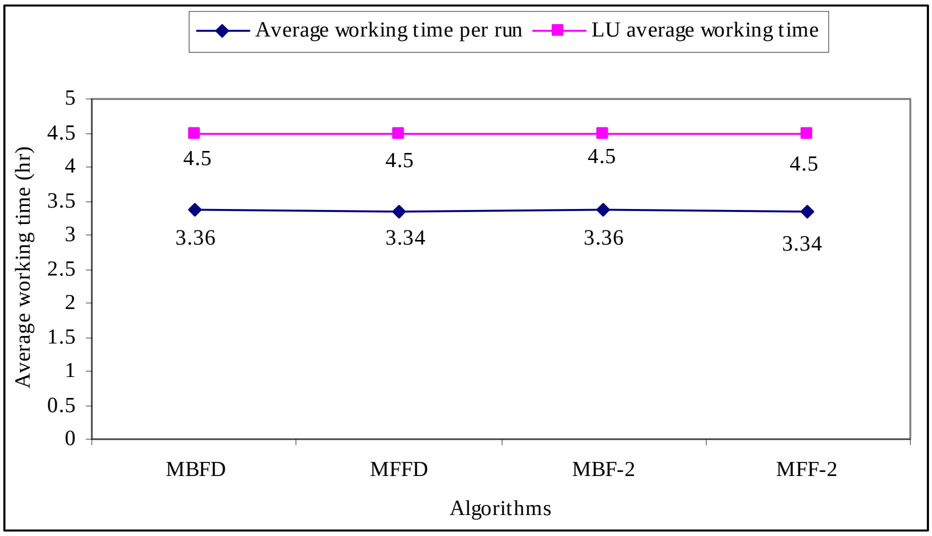


Figure 17 Comparison of average working time with LU criteria between algorithms for split blocks

Chapter 5: Conclusions and Directions for Future Study

This thesis proposed a new approach for solving the traditional crew scheduling problem. The crew scheduling problem solved with a bin packing approach is described along with its objectives and constraints. The CSP is decomposed and solved in two steps, a block cutting step and a piece matching step. The process introduction and solution methods for both steps are described in this study.

A comprehensive literature review was conducted both for the crew scheduling problem and the bin packing problem. A variety of algorithms and heuristics for this problem were described and analyzed. We concluded that the general steps for solving the crew scheduling problem are as follows: (1) generating an initial candidate set of feasible solutions, (2) selecting from the candidate set a minimum cost subset to cover all the tasks, (3) solving the SCP or SPP using appropriate algorithms.

The review of the literature indicated that heuristic algorithms are very popular in solving the NP hard crew scheduling problems since exhaustive search for optimal solution is usually computationally impossible, especially when the problem size grows large in the real world.

The bin packing problem, as a combinatorial problem in literature of operation research and mathematics, has been discussed for nearly 40 years and has an extensive application in the industry. Reviewing the heuristics approach proposed and applied in CSP indicated that solving CSP using a bin packing approach could be a fruitful area of research. In this study we conducted an extensive literature research in the original bin packing problem and considered the compatibility of the bin packing and the crew scheduling problems. We then formulated the piece matching process in CSP as a bin packing problem. To incorporate the bin packing feature into CSP, each run can be represented by a bin, and the pieces of work are items to be packed into bins. The associated working rules can be reflected by the capacity of the bin. Considering the similar natures of the BP and CSP, modifying and applying an efficient BP algorithm to CSP provided a faster heuristic to better, near optimal solution.

Based on the extensive research on bin packing problem solution approaches, an algorithm for solving most BPP (Best-Fit-Decreasing Algorithm) that proved to be the most efficient was selected in this study and modified for application to the crew scheduling problem.

In this study, CSP was decomposed into two parts. In the block cutting process, given the vehicle block information, all possible block partition results for each block were generated. In the piece matching process, a Modified Best-Fit-Decreasing Algorithm was proposed for solving the piece matching problem. This algorithm matches the pieces together to generate a set of runs covering all the tasks and performs in polynomial time. Test problems were solved in this study both through the mathematical model for the optimal solution and the proposed heuristic for near optimal solution.

A case study was conducted to test the proposed algorithm. This case study involved 73 vehicle blocks and 1299 tasks. The results obtained from the optimal solution and the heuristic solution were compared and discussed. The average working time of the runs was another performance measure that was selected to evaluate the results and it showed that the results are of good quality. Further test results indicated that when the problem size grows, optimal solution can not be obtained using CPLEX and the proposed heuristic can get the near optimal solution in seconds.

Incorporating the bin packing problem feature into crew scheduling problem is a new approach proposed in this study for solving CSP. Both bin packing and crew scheduling problem have been studied as classic combinatorial problems for decades, however, incorporating these two problems and applying an efficient solution algorithm from one to another has not be studied or discussed before. This study introduced the CSP with a bin packing feature and adopted the most efficient BP algorithm for application in CSP. The results show that the compatible features of these two problems made the modified BP heuristic also efficient for CSP.

In this proposed algorithm, all possible partitions are generated from the block cutting process, however, not every possible partition is included in the matching process. As a result, better piece matches may be missing during this process. Future study may focus on including as many different partitions as possible to generate a larger candidate piece set. By including more matching possibilities, a better result may be obtained.

In this study, the vehicle scheduling and crew scheduling are considered independently. Simultaneous vehicle and crew scheduling problem are studied and discussed extensively now. In this study, the vehicle blocks are given beforehand; however, an integrated approach for solving the vehicle and crew scheduling simultaneously may be considered to optimize the results.

Appendices

Appendix 1

Block 9:

Task number	Departing time	Arriving time	Relief point
1	706	718	TP
2	722	733	LP
3	742	811	SS
4	832	842	LP
5	849	913	SS
6	921	946	LP
7	949	1011	SS
8	1021	1046	LP
9	1049	1112	SS
10	1121	1146	LP
11	1149	1213	SS
12	1221	1248	LP
13	1251	1315	SS
14	1321	1348	LP
15	1351	1415	SS
16	1420	1448	LP
17	1451	1515	SS
18	1520	1548	LP
19	1551	1617	SS
20	1620	1650	LP
21	1653	1706	TP
22	1719	1742	TP
23	1806	1834	LP

Partition result for block 9:

3 4 6
0 426 553 561 553 561 795 801 795 801 1114 1114
DP SS SS SS SS LP
3 5 5
0 426 553 561 553 561 855 860 855 860 1114 1114
DP SS SS SS SS LP
3 6 4
0 426 553 561 553 561 915 920 915 920 1114 1114
DP SS SS SS SS LP
4 3 6
0 426 611 621 611 621 795 801 795 801 1114 1114
DP SS SS SS SS LP
4 4 5
0 426 611 621 611 621 855 860 855 860 1114 1114
DP SS SS SS SS LP
4 5 4
0 426 611 621 611 621 915 920 915 920 1114 1114
DP SS SS SS SS LP
4 6 3
0 426 611 621 611 621 977 980 977 980 1114 1114

DP SS SS SS SS LP
5 3 5
0 426 672 681 672 681 855 860 855 860 1114 1114
DP SS SS SS SS LP
5 4 4
0 426 672 681 672 681 915 920 915 920 1114 1114
DP SS SS SS SS LP
5 5 3
0 426 672 681 672 681 977 980 977 980 1114 1114
DP SS SS SS SS LP
6 3 4
0 426 733 741 733 741 915 920 915 920 1114 1114
DP SS SS SS SS LP
6 4 3
0 426 733 741 733 741 977 980 977 980 1114 1114
DP SS SS SS SS LP
3 3 3 4
0 426 553 561 553 561 733 741 733 741 915 920 915 920 1114 1114
DP SS SS SS SS SS SS LP
3 3 4 3
0 426 553 561 553 561 733 741 733 741 977 980 977 980 1114 1114
DP SS SS SS SS SS SS LP
3 4 3 3
0 426 553 561 553 561 795 801 795 801 977 980 977 980 1114 1114
DP SS SS SS SS SS SS LP
4 3 3 3
0 426 611 621 611 621 795 801 795 801 977 980 977 980 1114 1114
DP SS SS SS SS SS SS LP

Appendix 2

Block 10:

Task number	Departing time	Arriving time	Relief point
1	711	730	WH
2	738	752	FG
3	816	842	SS
4	851	916	LP
5	919	941	SS
6	951	1016	LP
7	1019	1041	SS
8	1051	1116	LP
9	1119	1143	SS
10	1151	1218	LP
11	1221	1245	SS
12	1251	1318	LP
13	1321	1345	SS
14	1351	1418	LP
15	1421	1445	SS
16	1453	1528	WH
17	1533	1608	SS
18	1622	1700	WH
19	1702	1740	SS
20	1742	1805	TP
21	1808	1822	LP
22	1833	1846	TP
23	1852	1906	LP

Partition results for block 10:

4 4 8
0 431 581 591 581 591 825 831 825 831 1146 1146
DP SS SS SS SS LP
4 5 7
0 431 581 591 581 591 885 893 885 893 1146 1146
DP SS SS SS SS LP
4 6 6
0 431 581 591 581 591 928 933 928 933 1146 1146
DP SS SS WH WH LP
5 3 8
0 431 641 651 641 651 825 831 825 831 1146 1146
DP SS SS SS SS LP
5 4 7
0 431 641 651 641 651 885 893 885 893 1146 1146
DP SS SS SS SS LP
5 5 6
0 431 641 651 641 651 928 933 928 933 1146 1146
DP SS SS WH WH LP
5 6 5
0 431 641 651 641 651 968 982 968 982 1146 1146
DP SS SS SS SS LP
6 3 7
0 431 703 711 703 711 885 893 885 893 1146 1146
DP SS SS SS SS LP
6 4 6

0 431 703 711 703 711 928 933 928 933 1146 1146
DP SS SS WH WH LP
6 5 5
0 431 703 711 703 711 968 982 968 982 1146 1146
DP SS SS SS SS LP
6 6 4
0 431 703 711 703 711 1020 1022 1020 1022 1146 1146
DP SS SS WH WH LP
7 3 6
0 431 765 771 765 771 928 933 928 933 1146 1146
DP SS SS WH WH LP
7 4 5
0 431 765 771 765 771 968 982 968 982 1146 1146
DP SS SS SS SS LP
7 5 4
0 431 765 771 765 771 1020 1022 1020 1022 1146 1146
DP SS SS WH WH LP
4 3 3 6
0 431 581 591 581 591 765 771 765 771 928 933 928 933 1146 1146
DP SS SS SS SS WH WH LP
4 3 4 5
0 431 581 591 581 591 765 771 765 771 968 982 968 982 1146 1146
DP SS SS SS SS SS SS LP
4 3 5 4
0 431 581 591 581 591 765 771 765 771 1020 1022 1020 1022 1146 1146
DP SS SS SS SS WH WH LP
4 4 3 5
0 431 581 591 581 591 825 831 825 831 968 982 968 982 1146 1146
DP SS SS SS SS SS SS LP
4 4 4 4
0 431 581 591 581 591 825 831 825 831 1020 1022 1020 1022 1146 1146
DP SS SS SS SS WH WH LP
4 5 3 4
0 431 581 591 581 591 885 893 885 893 1020 1022 1020 1022 1146 1146
DP SS SS SS SS WH WH LP
5 3 3 5
0 431 641 651 641 651 825 831 825 831 968 982 968 982 1146 1146
DP SS SS SS SS SS SS LP
5 3 4 4
0 431 641 651 641 651 825 831 825 831 1020 1022 1020 1022 1146 1146
DP SS SS SS SS WH WH LP
5 4 3 4
0 431 641 651 641 651 885 893 885 893 1020 1022 1020 1022 1146 1146
DP SS SS SS SS WH WH LP
6 3 3 4
0 431 703 711 703 711 885 893 885 893 1020 1022 1020 1022 1146 1146
DP SS SS SS SS WH WH LP

Appendix 3
Block1:

Task number	Departing time	Arriving time	Relief point
1	516	525	TP
2	532	541	LP
3	546	555	TP
4	602	612	LP
5	616	627	TP
6	632	642	LP
7	646	658	TP
8	705	729	SS
9	736	801	AK
10	807	833	SS
11	836	900	AK
12	910	933	SS
13	936	957	AK
14	1010	1033	SS
15	1036	1057	AK
16	1110	1133	SS
17	1136	1157	AK
18	1210	1233	SS
19	1236	1258	AK
20	1309	1333	SS
21	1336	1358	AK
22	1408	1433	SS
23	1450	1518	LP
24	1521	1547	SS
25	1550	1618	LP
26	1621	1648	SS
27	1652	1722	LP
28	1725	1738	TP
29	1744	1758	LP
30	1823	1859	WH
31	1903	1932	SS

Partition results for block 1:

5 6 6
0 316 449 456 449 456 813 816 813 816 1172 0
DP SS SS SS SS SS
6 5 6
0 316 513 516 513 516 813 816 813 816 1172 0
DP SS SS SS SS SS
6 6 5
0 316 513 516 513 516 873 890 873 890 1172 0
DP SS SS SS SS SS
7 4 6
0 316 573 576 573 576 813 816 813 816 1172 0
DP SS SS SS SS SS
7 5 5
0 316 573 576 573 576 873 890 873 890 1172 0

DP SS SS SS SS SS
8 3 6
0 316 633 636 633 636 813 816 813 816 1172 0
DP SS SS SS SS SS
8 4 5
0 316 633 636 633 636 873 890 873 890 1172 0
DP SS SS SS SS SS
8 5 4
0 316 633 636 633 636 947 950 947 950 1172 0
DP SS SS SS SS SS
5 3 3 6
0 316 449 456 449 456 633 636 633 636 813 816 813 816 1172 0
DP SS SS SS SS SS SS SS
5 3 4 5
0 316 449 456 449 456 633 636 633 636 873 890 873 890 1172 0
DP SS SS SS SS SS SS SS
5 3 5 4
0 316 449 456 449 456 633 636 633 636 947 950 947 950 1172 0
DP SS SS SS SS SS SS SS
5 4 3 5
0 316 449 456 449 456 693 696 693 696 873 890 873 890 1172 0
DP SS SS SS SS SS SS SS
5 4 4 4
0 316 449 456 449 456 693 696 693 696 947 950 947 950 1172 0
DP SS SS SS SS SS SS SS
5 4 5 3
0 316 449 456 449 456 693 696 693 696 1008 1012 1008 1012 1172 0
DP SS SS SS SS SS SS SS
5 5 3 4
0 316 449 456 449 456 753 756 753 756 947 950 947 950 1172 0
DP SS SS SS SS SS SS SS
5 5 4 3
0 316 449 456 449 456 753 756 753 756 1008 1012 1008 1012 1172 0
DP SS SS SS SS SS SS SS
5 6 2 4
0 316 449 456 449 456 813 816 813 816 947 950 947 950 1172 0
DP SS SS SS SS SS SS SS
5 6 3 3
0 316 449 456 449 456 813 816 813 816 1008 1012 1008 1012 1172 0
DP SS SS SS SS SS SS SS
6 3 3 5
0 316 513 516 513 516 693 696 693 696 873 890 873 890 1172 0
DP SS SS SS SS SS SS SS
6 3 4 4
0 316 513 516 513 516 693 696 693 696 947 950 947 950 1172 0
DP SS SS SS SS SS SS SS
6 3 5 3
0 316 513 516 513 516 693 696 693 696 1008 1012 1008 1012 1172 0
DP SS SS SS SS SS SS SS
6 4 3 4
0 316 513 516 513 516 753 756 753 756 947 950 947 950 1172 0
DP SS SS SS SS SS SS SS
6 4 4 3
0 316 513 516 513 516 753 756 753 756 1008 1012 1008 1012 1172 0
DP SS SS SS SS SS SS SS
6 5 2 4
0 316 513 516 513 516 813 816 813 816 947 950 947 950 1172 0
DP SS SS SS SS SS SS SS
6 5 3 3
0 316 513 516 513 516 813 816 813 816 1008 1012 1008 1012 1172 0
DP SS SS SS SS SS SS SS
7 3 3 4

0 316 573 576 573 576 753 756 753 756 947 950 947 950 1172 0
DP SS SS SS SS SS SS SS
7 3 4 3
0 316 573 576 573 576 753 756 753 756 1008 1012 1008 1012 1172 0
DP SS SS SS SS SS SS SS
7 4 2 4
0 316 573 576 573 576 813 816 813 816 947 950 947 950 1172 0
DP SS SS SS SS SS SS SS
7 4 3 3
0 316 573 576 573 576 813 816 813 816 1008 1012 1008 1012 1172 0
DP SS SS SS SS SS SS SS
8 3 2 4
0 316 633 636 633 636 813 816 813 816 947 950 947 950 1172 0
DP SS SS SS SS SS SS SS
8 3 3 3
0 316 633 636 633 636 813 816 813 816 1008 1012 1008 1012 1172 0
DP SS SS SS SS SS SS SS

Appendix 4
Block6:

Task number	Departing time	Arriving time	Relief point
1	638	706	AK
2	714	742	WH
3	803	838	SS
4	852	925	WH
5	930	1001	SS
6	1005	1026	TP
7	1031	1051	SS
8	1054	1126	WH
9	1130	1202	SS
10	1205	1226	TP
11	1231	1252	SS
12	1254	1327	WH
13	1330	1403	SS
14	1405	1428	TP
15	1432	1454	SS
16	1506	1529	AK
17	1537	1603	SS
18	1611	1644	HH
19	1655	1737	SS
20	1741	1815	HH
21	1825	1900	SS

Partition results for block 6:

2 9 6
0 398 518 532 518 532 807 810 807 810 1140 0
DP SS SS WH WH SS
2 10 5
0 398 518 532 518 532 843 845 843 845 1140 0
DP SS SS SS SS SS
2 11 4
0 398 518 532 518 532 868 872 868 872 1140 0
DP SS SS TP TP SS
3 8 6
0 398 565 570 565 570 807 810 807 810 1140 0
DP WH WH WH WH SS
3 9 5
0 398 565 570 565 570 843 845 843 845 1140 0
DP WH WH SS SS SS
3 10 4
0 398 565 570 565 570 868 872 868 872 1140 0
DP WH WH TP TP SS
3 11 3
0 398 565 570 565 570 894 906 894 906 1140 0
DP WH WH SS SS SS
4 7 6
0 398 601 605 601 605 807 810 807 810 1140 0
DP SS SS WH WH SS
4 8 5
0 398 601 605 601 605 843 845 843 845 1140 0

DP SS SS SS SS SS
4 9 4
0 398 601 605 601 605 868 872 868 872 1140 0
DP SS SS TP TP SS
4 10 3
0 398 601 605 601 605 894 906 894 906 1140 0
DP SS SS SS SS SS
4 11 2
0 398 601 605 601 605 963 971 963 971 1140 0
DP SS SS SS SS SS
5 6 6
0 398 626 631 626 631 807 810 807 810 1140 0
DP TP TP WH WH SS
5 7 5
0 398 626 631 626 631 843 845 843 845 1140 0
DP TP TP SS SS SS
5 8 4
0 398 626 631 626 631 868 872 868 872 1140 0
DP TP TP TP TP SS
5 9 3
0 398 626 631 626 631 894 906 894 906 1140 0
DP TP TP SS SS SS
5 10 2
0 398 626 631 626 631 963 971 963 971 1140 0
DP TP TP SS SS SS
6 5 6
0 398 651 654 651 654 807 810 807 810 1140 0
DP SS SS WH WH SS
6 6 5
0 398 651 654 651 654 843 845 843 845 1140 0
DP SS SS SS SS SS
6 7 4
0 398 651 654 651 654 868 872 868 872 1140 0
DP SS SS TP TP SS
6 8 3
0 398 651 654 651 654 894 906 894 906 1140 0
DP SS SS SS SS SS
6 9 2
0 398 651 654 651 654 963 971 963 971 1140 0
DP SS SS SS SS SS
7 5 5
0 398 686 690 686 690 843 845 843 845 1140 0
DP WH WH SS SS SS
7 6 4
0 398 686 690 686 690 868 872 868 872 1140 0
DP WH WH TP TP SS
7 7 3
0 398 686 690 686 690 894 906 894 906 1140 0
DP WH WH SS SS SS
7 8 2
0 398 686 690 686 690 963 971 963 971 1140 0
DP WH WH SS SS SS
8 5 4
0 398 722 725 722 725 868 872 868 872 1140 0
DP SS SS TP TP SS
8 6 3
0 398 722 725 722 725 894 906 894 906 1140 0
DP SS SS SS SS SS
8 7 2
0 398 722 725 722 725 963 971 963 971 1140 0
DP SS SS SS SS SS
9 5 3

0 398 746 751 746 751 894 906 894 906 1140 0
DP TP TP SS SS SS
9 6 2
0 398 746 751 746 751 963 971 963 971 1140 0
DP TP TP SS SS SS
2 5 5 5
0 398 518 532 518 532 686 690 686 690 843 845 843 845 1140 0
DP SS SS WH WH SS SS SS
2 5 6 4
0 398 518 532 518 532 686 690 686 690 868 872 868 872 1140 0
DP SS SS WH WH TP TP SS
2 5 7 3
0 398 518 532 518 532 686 690 686 690 894 906 894 906 1140 0
DP SS SS WH WH SS SS SS
2 5 8 2
0 398 518 532 518 532 686 690 686 690 963 971 963 971 1140 0
DP SS SS WH WH SS SS SS
2 6 5 4
0 398 518 532 518 532 722 725 722 725 868 872 868 872 1140 0
DP SS SS SS SS TP TP SS
2 6 6 3
0 398 518 532 518 532 722 725 722 725 894 906 894 906 1140 0
DP SS SS SS SS SS SS SS
2 6 7 2
0 398 518 532 518 532 722 725 722 725 963 971 963 971 1140 0
DP SS SS SS SS SS SS SS
2 7 5 3
0 398 518 532 518 532 746 751 746 751 894 906 894 906 1140 0
DP SS SS TP TP SS SS SS
2 7 6 2
0 398 518 532 518 532 746 751 746 751 963 971 963 971 1140 0
DP SS SS TP TP SS SS SS
2 8 4 3
0 398 518 532 518 532 772 774 772 774 894 906 894 906 1140 0
DP SS SS SS SS SS SS SS
2 8 5 2
0 398 518 532 518 532 772 774 772 774 963 971 963 971 1140 0
DP SS SS SS SS SS SS SS
2 9 4 2
0 398 518 532 518 532 807 810 807 810 963 971 963 971 1140 0
DP SS SS WH WH SS SS SS
3 5 5 4
0 398 565 570 565 570 722 725 722 725 868 872 868 872 1140 0
DP WH WH SS SS TP TP SS
3 5 6 3
0 398 565 570 565 570 722 725 722 725 894 906 894 906 1140 0
DP WH WH SS SS SS SS SS
3 5 7 2
0 398 565 570 565 570 722 725 722 725 963 971 963 971 1140 0
DP WH WH SS SS SS SS SS
3 6 5 3
0 398 565 570 565 570 746 751 746 751 894 906 894 906 1140 0
DP WH WH TP TP SS SS SS
3 6 6 2
0 398 565 570 565 570 746 751 746 751 963 971 963 971 1140 0
DP WH WH TP TP SS SS SS
3 7 4 3
0 398 565 570 565 570 772 774 772 774 894 906 894 906 1140 0
DP WH WH SS SS SS SS SS
3 7 5 2
0 398 565 570 565 570 772 774 772 774 963 971 963 971 1140 0
DP WH WH SS SS SS SS SS

3 8 4 2
0 398 565 570 565 570 807 810 807 810 963 971 963 971 1140 0
DP WH WH WH WH SS SS SS
4 5 5 3
0 398 601 605 601 605 746 751 746 751 894 906 894 906 1140 0
DP SS SS TP TP SS SS SS
4 5 6 2
0 398 601 605 601 605 746 751 746 751 963 971 963 971 1140 0
DP SS SS TP TP SS SS SS
4 6 4 3
0 398 601 605 601 605 772 774 772 774 894 906 894 906 1140 0
DP SS SS SS SS SS SS SS
4 6 5 2
0 398 601 605 601 605 772 774 772 774 963 971 963 971 1140 0
DP SS SS SS SS SS SS SS
4 7 4 2
0 398 601 605 601 605 807 810 807 810 963 971 963 971 1140 0
DP SS SS WH WH SS SS SS
5 5 4 3
0 398 626 631 626 631 772 774 772 774 894 906 894 906 1140 0
DP TP TP SS SS SS SS SS
5 5 5 2
0 398 626 631 626 631 772 774 772 774 963 971 963 971 1140 0
DP TP TP SS SS SS SS SS
5 6 4 2
0 398 626 631 626 631 807 810 807 810 963 971 963 971 1140 0
DP TP TP WH WH SS SS SS
6 5 4 2
0 398 651 654 651 654 807 810 807 810 963 971 963 971 1140 0
DP SS SS WH WH SS SS SS

Appendix 5

Block 7:

Task number	Departing time	Arriving time	Relief point
1	653	725	BS
2	734	816	GL
3	833	904	SS
4	924	956	WH
5	1000	1031	SS
6	1035	1056	TP
7	1101	1122	SS
8	1124	1156	WH
9	1200	1232	SS
10	1235	1256	TP
11	1301	1322	SS
12	1324	1358	WH
13	1403	1436	SS
14	1504	1548	GL
15	1558	1625	BC
16	1629	1654	GL
17	1658	1728	BC
18	1732	1803	GL
19	1808	1852	BS
20	1907	1936	FH
21	1943	1957	GE

Partition results of block 7:

3 9 6
0 413 544 564 544 564 838 843 838 843 1197 1197
DP SS SS WH WH GE
3 10 5
0 413 544 564 544 564 876 904 876 904 1197 1197
DP SS SS SS SS GE
4 8 6
0 413 596 600 596 600 838 843 838 843 1197 1197
DP WH WH WH WH GE
4 9 5
0 413 596 600 596 600 876 904 876 904 1197 1197
DP WH WH SS SS GE
4 10 4
0 413 596 600 596 600 948 958 948 958 1197 1197
DP WH WH GL GL GE
5 7 6
0 413 631 635 631 635 838 843 838 843 1197 1197
DP SS SS WH WH GE
5 8 5
0 413 631 635 631 635 876 904 876 904 1197 1197
DP SS SS SS SS GE
5 9 4
0 413 631 635 631 635 948 958 948 958 1197 1197
DP SS SS GL GL GE
6 6 6
0 413 656 661 656 661 838 843 838 843 1197 1197
DP TP TP WH WH GE
6 7 5

0 413 656 661 656 661 876 904 876 904 1197 1197
DP TP TP SS SS GE
6 8 4
0 413 656 661 656 661 948 958 948 958 1197 1197
DP TP TP GL GL GE
6 9 3
0 413 656 661 656 661 1014 1018 1014 1018 1197 1197
DP TP TP GL GL GE
7 5 6
0 413 682 684 682 684 838 843 838 843 1197 1197
DP SS SS WH WH GE
7 6 5
0 413 682 684 682 684 876 904 876 904 1197 1197
DP SS SS SS SS GE
7 7 4
0 413 682 684 682 684 948 958 948 958 1197 1197
DP SS SS GL GL GE
7 8 3
0 413 682 684 682 684 1014 1018 1014 1018 1197 1197
DP SS SS GL GL GE
8 5 5
0 413 716 720 716 720 876 904 876 904 1197 1197
DP WH WH SS SS GE
8 6 4
0 413 716 720 716 720 948 958 948 958 1197 1197
DP WH WH GL GL GE
8 7 3
0 413 716 720 716 720 1014 1018 1014 1018 1197 1197
DP WH WH GL GL GE
9 4 5
0 413 752 755 752 755 876 904 876 904 1197 1197
DP SS SS SS SS GE
9 5 4
0 413 752 755 752 755 948 958 948 958 1197 1197
DP SS SS GL GL GE
9 6 3
0 413 752 755 752 755 1014 1018 1014 1018 1197 1197
DP SS SS GL GL GE
3 5 5 5
0 413 544 564 544 564 716 720 716 720 876 904 876 904 1197 1197
DP SS SS WH WH SS SS GE
3 5 6 4
0 413 544 564 544 564 716 720 716 720 948 958 948 958 1197 1197
DP SS SS WH WH GL GL GE
3 5 7 3
0 413 544 564 544 564 716 720 716 720 1014 1018 1014 1018 1197 1197
DP SS SS WH WH GL GL GE
3 6 4 5
0 413 544 564 544 564 752 755 752 755 876 904 876 904 1197 1197
DP SS SS SS SS SS SS GE
3 6 5 4
0 413 544 564 544 564 752 755 752 755 948 958 948 958 1197 1197
DP SS SS SS SS GL GL GE
3 6 6 3
0 413 544 564 544 564 752 755 752 755 1014 1018 1014 1018 1197 1197
DP SS SS SS SS GL GL GE
3 7 4 4
0 413 544 564 544 564 776 781 776 781 948 958 948 958 1197 1197
DP SS SS TP TP GL GL GE
3 7 5 3
0 413 544 564 544 564 776 781 776 781 1014 1018 1014 1018 1197 1197
DP SS SS TP TP GL GL GE

3 8 3 4
0 413 544 564 544 564 802 804 802 804 948 958 948 958 1197 1197
DP SS SS SS SS GL GL GE
3 8 4 3
0 413 544 564 544 564 802 804 802 804 1014 1018 1014 1018 1197 1197
DP SS SS SS SS GL GL GE
3 9 3 3
0 413 544 564 544 564 838 843 838 843 1014 1018 1014 1018 1197 1197
DP SS SS WH WH GL GL GE
4 5 4 5
0 413 596 600 596 600 752 755 752 755 876 904 876 904 1197 1197
DP WH WH SS SS SS SS GE
4 5 5 4
0 413 596 600 596 600 752 755 752 755 948 958 948 958 1197 1197
DP WH WH SS SS GL GL GE
4 5 6 3
0 413 596 600 596 600 752 755 752 755 1014 1018 1014 1018 1197 1197
DP WH WH SS SS GL GL GE
4 6 4 4
0 413 596 600 596 600 776 781 776 781 948 958 948 958 1197 1197
DP WH WH TP TP GL GL GE
4 6 5 3
0 413 596 600 596 600 776 781 776 781 1014 1018 1014 1018 1197 1197
DP WH WH TP TP GL GL GE
4 7 3 4
0 413 596 600 596 600 802 804 802 804 948 958 948 958 1197 1197
DP WH WH SS SS GL GL GE
4 7 4 3
0 413 596 600 596 600 802 804 802 804 1014 1018 1014 1018 1197 1197
DP WH WH SS SS GL GL GE
4 8 3 3
0 413 596 600 596 600 838 843 838 843 1014 1018 1014 1018 1197 1197
DP WH WH WH WH GL GL GE
5 5 4 4
0 413 631 635 631 635 776 781 776 781 948 958 948 958 1197 1197
DP SS SS TP TP GL GL GE
5 5 5 3
0 413 631 635 631 635 776 781 776 781 1014 1018 1014 1018 1197 1197
DP SS SS TP TP GL GL GE
5 6 3 4
0 413 631 635 631 635 802 804 802 804 948 958 948 958 1197 1197
DP SS SS SS SS GL GL GE
5 6 4 3
0 413 631 635 631 635 802 804 802 804 1014 1018 1014 1018 1197 1197
DP SS SS SS SS GL GL GE
5 7 3 3
0 413 631 635 631 635 838 843 838 843 1014 1018 1014 1018 1197 1197
DP SS SS WH WH GL GL GE
6 5 3 4
0 413 656 661 656 661 802 804 802 804 948 958 948 958 1197 1197
DP TP TP SS SS GL GL GE
6 5 4 3
0 413 656 661 656 661 802 804 802 804 1014 1018 1014 1018 1197 1197
DP TP TP SS SS GL GL GE
6 6 3 3
0 413 656 661 656 661 838 843 838 843 1014 1018 1014 1018 1197 1197
DP TP TP WH WH GL GL GE
7 5 3 3
0 413 682 684 682 684 838 843 838 843 1014 1018 1014 1018 1197 1197
DP SS SS WH WH GL GL GE

Glossary

Block: A block is a sequence of tasks assigned to one bus for one day's work; it is a vehicle assignment

Run: A run is the work performed by a single crew for one single day; it is a crew assignment;

Task: A task is the trip between two relief points on a block; it is part of a piece of work.

Piece of work: A piece of work is a continuous work period composed by one or more consecutive tasks from a block covered by the same crew; it is a part of a run.

Route: A sequence of links (with stops) served by a single vehicle.

Trip: A single vehicle traveling through a route with no stop, which is the basic unit of service.

Relief Point: Stops along the route where a crew can take a break and another crew can take over the bus.

Depot: Parking and service location for vehicles when not required for service.

Partition: The set of pieces of work formed by making cuts at a subset of relief points on the vehicle block.

Meal Break: A rest period for a crew to get off the bus and have a meal.

Straight Run: A run with no meal break between the pieces of work, if there is a meal break, it will be paid. It is a continuous run.

Split Run: A run which has a long meal break between the pieces of work which may be unpaid. It is usually a two piece run with long break in the middle.

Tripper: A small piece of work (one or two consecutive tasks) which is unassigned to a crew and usually performed as overtime. It is a short run.

Spread time: The total time from the start to the end of the run.

Compatible Trip: A set of consecutive trips on a vehicle block that the second one can run after the first one by the same vehicle.

Dead-head time: Time that a vehicle is not generating money.

Layover time: Idle time before or after a trip.

Bibliography

Ball, M.O. and Benoit-Thompson, H. (1987). "A Lagrangian Relaxation Based heuristic for the Urban Transit Crew Scheduling Problwm." Proceedings of the Fourth International Workshop on Computer-Aided Scheduling of Public Transport. Hamburg, Germany.

Ball, M.O., Bodin, L. and Dial, R. (1983). "A matching based heuristic for scheduling mass transit crews and vehicles." *Transportation Science,* Vol.17, No.1.

Banihashemi, M. and Haghani, A. (2001). "A new model for the mass transit crew scheduling problem." In Computer-aided Scheduling of Public Transport. Lecture notes in Economics and Mathematical Systems, 505 (Voss, S. and Daduna, J. R. eds), pp.1-15, Springer-Verlag.

Beasley, J.E. and Cao, B. (1996). "A tree search algorithm for the crew scheduling problem." *European Journal of Operational Research*, 94,517-526.

Blais, J. Y. and Rousseau, J.M. (1987). "Overview of HASTUS Current and Future Versions" Proceedings of the Fourth International Workshop on Computer-Aided Scheduling of Public Transport. Hamburg, Germany.

Bodin, L.D., Golden, B., Assad A., Ball, M.O. (1983). "Routing and Scheduling of Vehicles and Crews: The State of the Art." *Computers and Operations Research* 10, 63-211.

Carraresi, P., Nonato, M. and Girard, L. (1995). "Network models, Lagrangean relaxation and subgradients bundle approach in crew scheduling problems." In Computer-aided Transit Scheduling. Lecture notes in Economics and Mathematical Systems, 430 (Daduna, J.R., Branco, I. and Paixao, J.M.P. eds), pp.188-212, Springer-Verlag.

Cavique, L., Rego, C. and Themido, I. (1999). "Subgraph ejection chains and tabu search for the crew scheduling problem." *Journal of the Operational Research Society* 50, 608-616.

Clement, R. and Wren, A. (1995). "Greedy genetic algorithms, optimizing mutations and bus driver scheduling." In Computer-aided Transit Scheduling. Lecture notes in Economics and Mathematical Systems, 430 (Daduna, J.R., Branco, I. and Paixao, J.M.P. eds), pp.213-235, Springer-Verlag.

Coffman, E.G., (ed.). (1976). Computer and Job-Shop Scheduling Theory. Wiley, New York.

Coffman, E.G., Garey, M.R. and Johnson, D.S. (1996). "Approximation algorithms for bin packing: a survey." In Hochbaum, D. (ed.). Approximation algorithms for NP-hard problems, pages 46–93. PWS Publishing, Boston.

Coffman, E.G., Leung, J.Y. and Ting, D.W. (1978). "Bin Packing: Maximizing the number of Pieces Packed." *Acta Inform.* 9, 263-271.

Daduna, J.R and Wren, A. (eds). (1987). Computer-Aided Transit Scheduling-Proceedings of the Fourth International Workshop on Computer-Aided Scheduling of Public Transport. Hamburg, Germany.

Daduna, J.R., Branco I., Paixao, J.M.P. (eds). (1993). Computer-Aided Transit Scheduling-Proceedings of the Sixth International Workshop on Computer-Aided Scheduling of Public Transport. Lisbon, Portugal.

Dantzig, G.B. and Wolfe, P.(1960). "Decomposition principles for linear programming." *Operations Research*, 8, 101–111.

Darby-Dowman, K., Jachnik, J.K., Lewis, R.L., and Mitra, G. (1988). "Integrated Decision Support Systems for Urban Transport Scheduling: Discussion of Implementation and Experience." Pages 226–239 of: Daduna, J.R., and Wren, A. (eds), Computer-Aided Transit Scheduling: Proceedings of the Fourth International Workshop. Springer Verlag, Berlin.

Desrochers, M. and Rousseau J.M. (eds). (1990). Computer-Aided Transit Scheduling-Proceedings of the Fifth International Workshop on computer-Aided Scheduling of Public Transport. Montreal, Canada.

Desrochers, M. and Soumis, F. (1989). "A column generation approach to the urban transit crew scheduling problem." *Transportation Science,* 23, l-13.

Edmonds, J. (1965). "Paths, Trees and Flowers," *Canad. J. Math.* 17, 440-467.

Falkner, J.C. and Ryan, D.M. (1987). "A bus crew scheduling system using a set partitioning model." *Asia-Pacific Journal of Operational Research* 4, 39-56.

Fischetti, M., Lodi, A., Martello, S. and Toth, P. (2001). "A Polyhedral Approach to Simplified Crew Scheduling and Vehicle Scheduling Problems." *Management Science*, Vol. 47, No. 6, pp.833-850.

Fischetti, M., Martello, S. Toth, P. (1989). "The fixed job schedule problem with working-time constraints." *Oper. Res.* 37, 395-403.

Fisher, M.L. (1980). "Worst-case Analysis of Heuristic Algorithms," *Management Science*, Vol. 26, No. 1.

Freling, R., Huisman, D., and Wagelmans, A.P.M. (2003). "Models and Algorithms for Integration of Vehicle and Crew Scheduling." *Journal of Scheduling,* 6, 63–85.

Freling, R., Wagelmans, A.P.M., and Paixao, J.M.P. (1999). "An Overview of Models and Techniques for Integrating Vehicle and Crew Scheduling." Pages 441–460 of: Wilson, N.H.M. (ed), Computer-Aided Transit Scheduling. Springer Verlag, Berlin.

Gaffi, A. and Nonato M. (1997). "An integrated Approach to Ex-Urban Crew and Vehicle Scheduling." Proceedings of the Seventh International Workshop on Computer-Aided Scheduling of Public Transport. Cambridge, MA

Gaffi, A., and Nonato, M. (1999). “An Integrated Approach to Extra-Urban Crew and Vehicle Scheduling.” Pages 103–128 of: Wilson, N.H.M. (ed), Computer-Aided Transit Scheduling. Springer Verlag, Berlin.

Haase, K., Desaulniers, G., and Desrosiers, J. (2001). “Simultaneous Vehicle and Crew Scheduling in Urban Mass Transit Systems.” *Transportation Science*, 35, 286–303.

Hartley, T. (1981). “A glossary of terms in bus and crew scheduling.” In: Wren, A. (ed). Computer Scheduling of Public Transport Urban Passenger Vehicle and Crew Scheduling-Proceedings of the Second International Workshop on Computer-Aided Scheduling of Public Transport. Leeds, England.

Heurgon, E. (1975). “Preparing Duty Roster for bus Routes by Computer,” Preprint Workshop on Automated Technique for Scheduling of Vehicle Operators for Urban Public Transportation Services, Chicago.

Huisman, D., Freling, R., Albert, P., Wagelmans, M. (2005). “Multiple-Depot Integrated Vehicle and Crew Scheduling.” *Transportation Science*, Vol. 39, No. 4, pp. 491-502.

Johnson, D. S. (1974). “Past Algorithms for Bin-Packing.” *J. Comput. Syst. Sci*. 8, 272-314.

Johnson, D. S., Demers, A., Ullman, J.D., Garey, M.R. and Graham, R.L. (1974). “Worst-Case Performance Bounds for Simple One-Dimensional Packing Algorithms.” *SIAM J.Comput.* 3, 299-326.

Li, J. and Kwan, R.S.K. (2003). “A fuzzy genetic algorithm for driver scheduling.” *European Journal of Operational Research* 147, 334 – 344.

Lourenco, H. R., Paixao, J.P. and Portugal, R. (2001). “Multiobjective Metaheuristics for the Bus Driver Scheduling Problem,” *Transportation Science*, 35(3), 331-343.

Martello S. and Toth P., (1990). “Knapsack Problems, Algorithms and Computer implementations.” John Wiley and Sons Ltd: England.

Martello, S. and Toth, P. (1986). “A heuristic approach to the bus driver scheduling problem.” *European Journal of Operational Research* 24, 106- 117.

Mitra, G. and Darby-Dowman, K. (1985). “CRU-SCHED: A computer-based bus crew scheduling system using integer programming.” In Computer Scheduling of Public Transport-2, J. M. Rousseau, ed., pp. 223-232.

Ong, H.L, Magazine, M.J. and Wee, T. S. (1984). “Probabilistic Analysis of Bin Packing heuristics.” *Operations Research,* Vol.32, No.5.

Paias, A. and Paixao, J. M. P. (1993). “State space relaxation for set covering problems related to bus driver scheduling.” *European Journal of Operational Research*, 71, 303-316.

Paixao, J. M. P. (1990). “Transit crew scheduling on a personal workstation.” *Operational Research,* 90 (Bradley, H. ed.), pp. 421-432, Pergamon Press.

Rousseau, J.M. (ed). (1983). Computer Scheduling of Public Transport -Proceedings of the Third International Workshop on Computer-Aided Scheduling of Public Transport. Montreal, Canada.

Shepardson, F. and Marsten, R. E. (1980). “A Lagrangean Relaxation Algorithm for the Two Duty Period Scheduling Problem,” *Mgmt. Sci*. 26, 274-281.

Smith, B.M. and Wren, A. (1988). “A bus crew scheduling system using a set covering formulation.” *Transportation Research*, 22A, 97-108.

Voss, S. and Daduna, J. R. (eds) (2000). Computer-Aided Transit Scheduling-Proceedings of the Eighth International Workshop on computer-Aided Scheduling of Public Transport. Berlin, Germany.

Wilson. N.H.M. (ed). (1997). Computer-Aided Transit Scheduling-Proceedings of the Seventh International Workshop on Computer-Aided Scheduling of Public Transport. Cambridge, MA

Wren, A. (ed). (1980). Computer Scheduling of Public Transport Urban Passenger Vehicle and Crew Scheduling-Proceedings of the Second International Workshop on Computer-Aided Scheduling of Public Transport. Leeds, England.

Wren, A. and Rousseau, J.M. (1995). “Bus driver scheduling-an overview.” In Daduna, J. R., Branco, I., and Paixao, J.M.P. (eds). Computer-aided Transit Scheduling, pages 173-187. Springer-Verlag.

Wren, A., Smith, B.M. and Miller, A.J. (1985). “Complementary Approaches to Crew Scheduling,” in Rousseau, J.M. (ed.). Computer Scheduling of Public Transport 2, Elsevier, Amsterdam.