Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Subdirección de Estudios de Posgrado

Lot Production Size Problem and Simulation of Urban Transport

por

M.C. Miguel Lorenzo Morales Marroquín

en opción al grado de

Doctor en Ingeniería

con especialidad en Ingeniería de Sistemas

San Nicolás de los Garza, Nuevo León, diciembre 2015

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Subdirección de Estudios de Posgrado

Lot Production Size Problem and Simulation of Urban Transport

por

M.C. Miguel Lorenzo Morales Marroquín

en opción al grado de

Doctor en Ingeniería

con especialidad en Ingeniería de Sistemas

San Nicolás de los Garza, Nuevo León, diciembre 2015

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Subdirección de Estudios de Posgrado

Los miembros del Comité de Tesis recomendamos que la Tesis "Lot Production Size Problem and Simulation of Urban Transport", realizada por el alumno M.C. Miguel Lorenzo Morales Marroquín, con número de matrícula 1022145, sea aceptada para su defensa como requisito parcial para obtener el grado de Doctor en Ingeniería con especialidad en ingeniería de sistemas.

El Comité de Tesis

[illegible]

Dra. Yasmín Á. Ríos Solís

Directora

Dr. Edgar Possani Espinosa

Revisor

Dra. Marta Cabo Nodar

Revisora

[illegible]

Dr. Romeo Sánchez Nigenda

Revisor

[illegible]

Dr. Igor S. litvinchev

Revisor

Vo. Bo.

[illegible]

Dr. Simón Martínez Martínez

Subdirección de Estudios de Posgrado

San Nicolás de los Garza, Nuevo León, diciembre 2015

To my wife who is the mainstay of all my projects. Without her this work would not have been possible.

To my daughters, Regina and Minerva, that gave up some of their playing time with his dad for me to finish this dream.

Acknowledgements

I wish thank Genesis for her understanding and support during this period. My daughters Regina and Minerva gave me strength everyday. My parents Miguel and Blanca provided me their best advice and always have been there when I need them more.

I with to thank my classmates. We spent so many nice hours together during my time studying the PhD.

My greatest appreciation and deep admiration to all my teachers for their advice and knowledge transmitted, especially to Yasmin A. Ríos Solís for her continuous support in my development as a researcher and her excellent advice all along this PhD thesis.

To my reviewers Igor S. Litvinchev, Romeo Sanchez Nigenda, Marta Cabo Nodar, and Edgar Possani Espinosa, I thank you in a very special way for being part of this dissertation committee, for their advice and total readiness and accessibility throughout this PhD Thesis.

I acknowledge the support FIME and CONACyT that provided my grants that where fundamental to the realization of this dissertation.

CONTENTS

ABSTRACT

M.C. Miguel Lorenzo Morales Marroquín.

Candidato para obtener el grado de Doctor en Ingeniería con especialidad en ingeniería de sistemas.

Universidad Autónoma de Nuevo León.

Facultad de Ingeniería Mecánica y Eléctrica.

Título del estudio: LOT PRODUCTION SIZE PROBLEM AND SIMULATION OF URBAN TRANSPORT.

Número de páginas: 65.

OBJECTIVES AND STUDY METHOD: There are two subjects in this thesis: “Lot production size for a parallel machine scheduling problem with auxiliary equipment” and “Bus holding for a simulated traffic network”. Although these two themes seem unrelated, the main idea is the optimization of complex systems.

The “Lot production size for a parallel machine scheduling problem with auxiliary equipment” deals with a manufacturing setting where sets of pieces form finished products. The aim is to maximize the profit of the finished products. Each piece may be processed in more than one mold. Molds must be mounted on machines with their corresponding installation setup times. The key point of our methodology is to solve the single period lot-sizing decisions for the finished products together with the piece-mold and the mold-machine assignments, relaxing the constraint that a single mold may not be used in two machines at the same time.

For the "Bus holding for a simulated traffic network" we deal with One of the most annoying problems in urban bus operations is bus bunching, which happens when two or more buses arrive at a stop nose to tail. Bus bunching reflects an unreliable service that affects transit operations by increasing passenger-waiting times. This work proposes a linear mathematical programming model that establishes bus holding times at certain stops along a transit corridor to avoid bus bunching. Our approach needs real-time input, so we simulate a transit corridor and apply our mathematical model to the data generated. Thus, the inherent variability of a transit system is considered by the simulation, while the optimization model takes into account the key variables and constraints of the bus operation.

CONTRIBUTIONS AND CONCLUSIONS: For the "Lot production size for a parallel machine scheduling problem with auxiliary equipment" the relaxation we propose able to find solutions more efficiently, moreover our experimental results show that most of the solutions verify that molds are non-overlapping even if they are installed on several machines. We propose an exact integer linear programming, a Relax&Fix heuristic, and a multistart greedy algorithm to solve this problem. Experimental results on instances based on real-world data show the efficiency of our approaches. The mathematical model and the algorithm for the lot production size problem, showed in this research, can be used for production planners to help in the scheduling of the manufacturing.

For the "Bus holding for a simulated traffic network" most of the literature considers quadratic models that minimize passenger-waiting times, but they are harder to solve and therefore difficult to operate by real-time systems. On the other hand, our methodology reduces passenger-waiting times efficiently given our linear programming model, with the characteristic of applying control intervals just every 5 minutes.

Firma de la directora: ____________________________

Dra. Yasmín Á. Ríos Solís

CHAPTER 1

INTRODUCTION

In this chapter we will give a short introduction of the subjects that we cover on this dissertation. The first one is related to the "Lot production size for a parallel machine scheduling problem with auxiliary equipment" that we describe in Section 1.1 while the second subject is the "Bus holding for a simulated traffic network" that is presented in Section 1.2. Although these two themes seem unrelated, the main idea is the optimization of complex systems. Finally, in Section 1.3 we introduce the dissertation structure.

1.1 LOT PRODUCTION SIZE FOR A PARALLEL MACHINE SCHEDULING PROBLEM WITH AUXILIARY EQUIPMENT

1.1.1 PROBLEM STATEMENT

The problem under study is inspired by a manufacturing company that produces plastic products. Plastic injection systems use molds for shaping raw plastic into pieces that are then assembled to form final products. Molds that produce a certain piece may have different production rates, and each machine has its own production speed. The objective of the company is to determine the lot-size of finished products by making an accurate assignment of the production pieces to molds and molds to machines in order to manufacture the finished products.

These products are composed by some pieces that are completely characterized by their shape, color, and plastic type. Notice that some pieces are shared by more than one product, while others are exclusive of a single one. Each mold may be different due to its technical specifications; of particular relevance is the type and number of cavities. Even though a mold may produce different pieces, once a piece is selected no other piece may be produced with the same mold at the same time. Finally, the molds must be mounted on machines. Note that not all molds may be installed on any machine.

In this problem we are only interested in finished products (those that the company is able to sell). We only consider a single period, hence inventory is not taken into account. The demand for the plastic products is seldom fulfilled by the company. Thus, the production capacity is expected to be fully used.

1.1.2 OBJECTIVES

To develop a methodology that helps the decision maker to find a better solution for this problem in an efficient time. The main idea is to maximize the profits of the factory, finding a combination of products to be made with the resources, machines and auxiliary equipment, in the best possible manner.

1.1.3 HYPOTHESIS

The development of an operations research methodology will help decision maker to find a solution in a more efficient way by finding better quality solutions that improve the profits of the factory, while taking advantage of the the factory resources.

1.1.4 SCIENTIFIC METHODOLOGY

The following methodology is used to achieve the objective previously proposed:

- Statement of the problem. This stage is to state the problem and define all the peculiarities of the problem while delimiting its scope.
- Bibliography review. This stage is to find similar problems and different methodologies proposed to solve it.
- Mathematical formulation. In here we developed a integer programming model that fits all the peculiarities of the problem to solve it.
- Experimentation for Mathematical formulation. In here we develop an instances generator for the problem and use them to solve it with GAMS/CPLEX.
- Result analysis of the mathematical formulation. In this stage we analyze the efficiency of the results of the mathematical model solved with GAMS/CPLEX.
- Heuristic implementation. Here we propose the use of a heuristic method to solve the problem.
- Experimentation of heuristic alternatives. In this stage we implement different heuristic methods to solve the problem.
- Result analysis for the different heuristic alternatives.

1.2 BUS HOLDING FOR A SIMULATED TRAFFIC NETWORK

1.2.1 PROBLEM STATEMENT

The study of complex bus operating systems is usually divided in two main areas, line planning and real-time control. The line planning process involves strategic, tactical and operational decisions. Strategic problems relate to long-term network design decisions. Tactical and operational decisions ultimately define the service offered to the public; for example, frequency of buses, definition of stops, bus timetabling, vehicle scheduling, driver scheduling, maintenance scheduling, among other problems. On the other hand, real-time control tries to maintain the bus system operational along the day in order to minimize passenger inconvenience.

Real-time control is hard because of the dynamics of the network or trattic situations. Although, bus frequency is planned for each stop in the network, changes in the passenger flow, traffic or even in the timetabling, produce perturbations that alter frequency plans and give rise to one of the most annoying problems in urban transportation operations, the bus bunching problem that happens when two or more buses arrive at a stop nose to tail, and it reflects an unreliable service that affects transit operations by increasing passenger-waiting times.

On this work, we focus on providing solutions to the bus bunching problem by maintaining congruent headways. Furthermore, we will show that maintaining congruent headways implicitly reduces passenger-waiting times. A headway is a quality measure given to the time difference between two consecutive buses. A bus line could have equally distant headways or different ones for each pair of buses.

1.2.2 OBJECTIVES

Most of the works in the literature base their quality measure on the waiting times of the passengers, or the variance between the departure times of the buses, which can be modeled with quadratic functions that are harder to solve. Instead, by using a linear programming model, our methodology returns optimal solutions at a fraction of the cost of alternative approaches. Moreover, maintaining more stable headways reduces passenger-waiting times in the lines, as our experimental results will demonstrate.

1.2.3 HYPOTHESIS

The development of an efficient methodology to help regulate the transport lines to the original planning will improve the experience offered to the transport end users.

1.2.4 SCIENTIFIC METHODOLOGY

- Statement of the problem. In this stage we present the problem and we will define all its peculiarities and its scope.
- Bibliography review. Here we look for different approaches to resolve similar problems and we analyze the different proposals for solving and their analysis methods.
- Mathematical formulation. In this stage we propose a lineal programming model that adjusts to all the peculiarities of the problem, to obtain an action plan to regulate the transport line.
- Simulation approach. Here we propose a simulation, designed in ExtendSim, that adjusts to the peculiarities of the problem.
- Experimentation. In this stage we use the simulation previously proposed to generate the different instances to be solved by the lineal programming modeled with GUROBI.
- Result analysis. In here we will analyze the result obtained in the simulation implementing the lineal programming model.

1.3 DISSERTATION STRUCTURE

In this section we describe the structure of this dissertation, which is divided into the two main subjects as it is explained below.

- Chapter 2, presents the research about the "Lot production size for a parallel machine scheduling problem with auxiliary equipment". Section 2.1 is the summary of this subject while in Section 2.2 we show the description of the problem. In Section 2.3 we present a new integer linear programming for the PPMM problem. Then, in Section 2.4, we develop a Relax&Fix heuristic, while in Section 2.5 we solve the problem with a multistart greedy algorithm. In Section 2.6 we show that our iterative procedure is indeed efficient since the solutions obtained by the heuristic are of high quality and have feasible mold scheduling solutions. Finally, in Section 2.7 we conclude and point out further areas of research.

- Chapter 3 presents the research about the "Linear Bus Holding Model for Real Time Traffic Network Control". In Section 3.1 we present a abstract for this research while in Section 3.2 we describe the problem. A brief revision of the state of the art is presented in Section 3.3. In Section 3.4, we present our new linear programming model inspired in earliness and tardiness penalties of just-in- time scheduling problems, which determines the optimal holding times of the buses at the stops. Then, Section 3.5 shows the efficiency of our model on a discrete event simulation of a single corridor. Finally, Section 3.6 presents our conclusions, and discusses open research questions that arise from this work.

- In Chapter 4 we present the general conclusions of this dissertation. We also talk about the future work that emerged from this investigation.

CHAPTER 2

LOT PRODUCTION SIZE FOR A PARALLEL MACHINE SCHEDULING PROBLEM WITH AUXILIARY EQUIPMENT

This research produced the article:

> Yasmín A. Ríos-Solís, Miguel L. Morales-Marroquín, Marta Cabo, Edgar Possani.“Lot production size for a parallel machine scheduling problem with auxiliary equipment”. Submitted to Annals of Operations Research.

2.1 ABSTRACT

We study a manufacturing setting where sets of pieces form finished products, each piece has a set of molds that can be used to make it, and molds must be mounted on machines with their corresponding installation setup times. The aim is to maximize the profit of the finished products. The key point of our methodology is to solve the single period lot-sizing decisions for the finished products together with the piece-mold assignments and the mold-machine ones, but without the mold overlapping constraints on different machines. We propose an exact integer linear programming, a Relax&Fix heuristic, and

a multistart greedy algorithm to solve this problem. Experimental results on instances based on real-world data show the efficiency of our approaches and exhibit that most of the solutions verify that molds are non-overlapping even if they are installed on several machines.

2.2 INTRODUCTION

The problem we study in this paper is inspired by a manufacturing company that produces plastic products. Plastic injection systems molds are used for shaping raw plastic into pieces that are then assembled to form final products. Molds that produce a certain piece may have different production rates (or different cavities), and each machine has its own production speed. The objective of the company is to determine the lot-size of finished products by making an accurate assignment of the production pieces to molds and machines to manufacture the finished products. We name this problem the Product-Piece-Mold-Machine problem (the PPMM problem for short).

Figure 2.1 illustrates most of the characteristics of the PPMM problem. The first column shows the finished products (male doll, female doll, and robot). These products are composed by some of the pieces in the second column of the figure. A piece is completely characterized by its shape, color, and plastic type. An edge between a product and a piece indicates that this piece is needed for assembling this product. Notice that some pieces such as the arms, are needed to assemble the male and the female dolls. Some other pieces such as the dress of the female doll, are exclusive of a single product. Edges between a piece and a mold (second to third column) indicate that this piece can be manufactured with that mold. Each mold may be different due to technical specifications (material and number of cavities). Even though a mold may produce different pieces, once a piece is selected no other piece may be produced with this same mold at the same time. That is, the first mold can produce either faces or arms at each time. Finally, the last column represent the machines where the molds are installed. Note that not all molds may be installed on any machine.

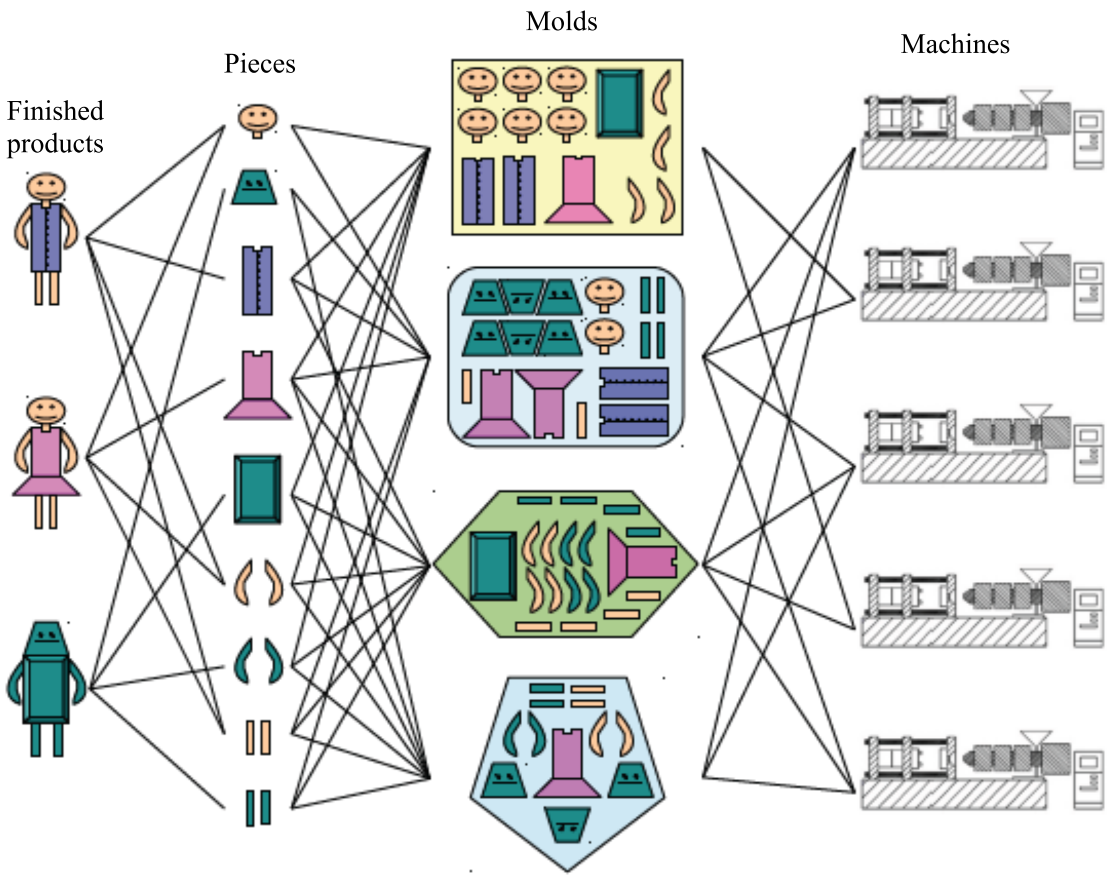

Figure 2.1: Scheme of the PPMM problem: sets of pieces form finished products, each piece has a set of molds that can be used to make it, and molds are mounted on a certain number of machines

When the production of a specific type of piece on some machine is finished, there are two possible decisions: to process the next piece with the same mold, or to remove the mold from this machine. In the first case, a small preparation time may arise. This setup time between the production of two pieces may be a color or a plastic change. In the other case, if another mold is going to be installed on this machine, its installation and removal incur in a *mold-setup time* that is larger than the piece setup time. Molds are usually heavy and need to be moved by cranes in order to be installed on the machines. Therefore, a large amount of time is used for their installation, preparation, and removal (two hours on average for the plastic injection molds).

In Figure 2.2 a feasible solution of the PPMM system is represented. On Machine 1 the first mold installed is Mold 1 that first produces Piece 1 and then Piece 6 with a setup time between these two pieces. After that, a larger setup time is incurred when changing to Mold 2 on the same machine. Notice that Mold 1 is also used on Machine 2 but not at the same time. The key point of our approach is that in the PPMM problem we drop the constraints that guarantee that a mold is not used on more than one machine at a time. With this modification we obtain a more tractable problem that has in most of the cases a feasible scheduling of the molds as shown in Section 2.6. Note that a piece may be produced at the same time in two different machines using different molds as Piece 2 in Mold 4 on Machine 2, and in Mold 3 on Machine 3.

In the PPMM environment the company can only sell finished products. That is, we are considering a two-level lot-sizing problem that corresponds to an assembly structure. The demand of the plastic finished products is seldom fulfilled by the company. Thus, the production capacity is expected to be fully used. In this PPMM setting, only a single period is considered, that is, no inventory is allowed.

In the literature, scheduling problems often consider sequence dependent setup times between the processing of different types of jobs (here, molds) as in PPMM. Most of these works consider the single machine case (Eren, 2007; Sourd, 2006; Stecco *et al.*, 2008). Studies that deal with the parallel machine case and sequence dependent setup times, for

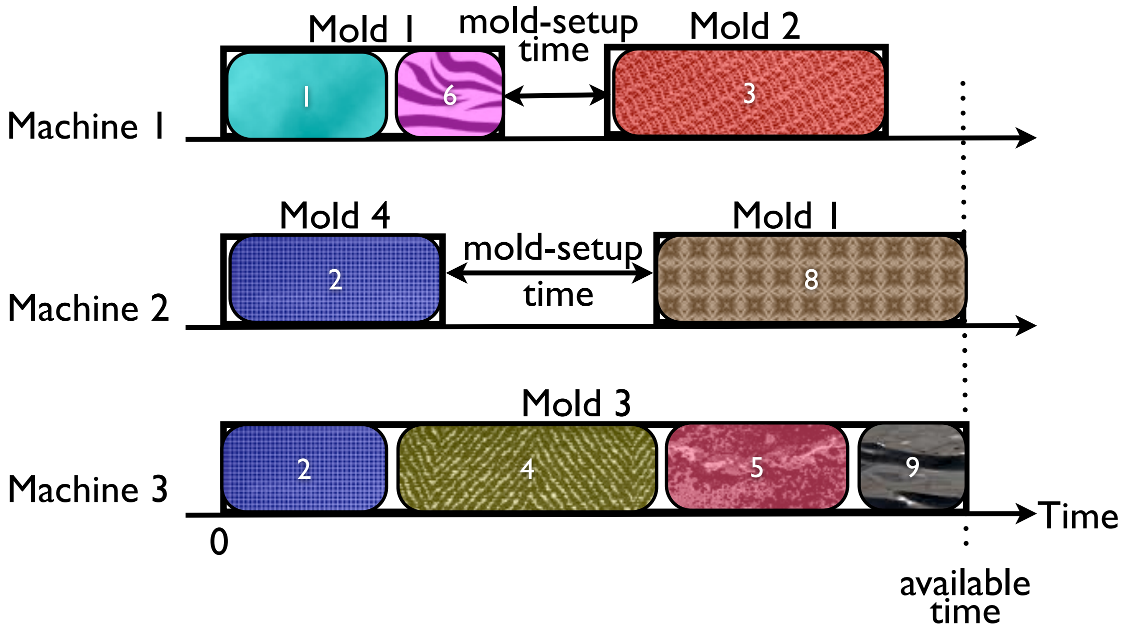

Figure 2.2: Gantt chart of the PPMM production

example Weng *et al.* (2001), do not consider the additional complexity added by using auxiliary equipment such as the molds in our PPMM problem. A recurrent idea it to base the scheduling methodologies on scheduling families of jobs (Chen y Powell, 2003; Haase y Kimms, 2000; Li *et al.*, 2005; Vilím, 2006). Marinelli *et al.* (2007) propose heuristics for the integrated lot-sizing and scheduling problems. Again, none of these models consider auxiliary production equipment.

In a classical approach to solve the PPMM problem, first the lot-sizing problem would be solved and then the mold-machine and piece-mold assignments would be tackled (Quadt y Kuhn, 2008; Gicquel *et al.*, 2008). Some other works integrate the batch sizing and scheduling problems for a single machine (Chrétienne *et al.*, 2011; Hazır y Kedad-Sidhoum, 2012). In Ibarra-Rojas *et al.* (2011), the authors propose a decomposition approach for a problem similar to the PPMM problem except that they do not consider that the pieces must form finished products. In Section 2.3 we show that the structure of our integer linear programming is drastically different from theirs because we consider finished products. So neither their model nor their iterated local search heuristic can be easily adapted to the PPMM problem, to the best of our knowledge.

The PPMM problem is innovative in three ways. The first one is that we do not make assumptions about the auxiliary equipment such as unique machine assignments as it is done in Lin *et al.* (2002), Ibarra-Rojas *et al.* (2010), and in Chacón Mondragón *et al.* (2012). When molds are forced to be installed in only one machine this implies sub-optimal solutions since it does not consider the possibility of processing pieces with a specific mold on different machines, which is a feasible solution and often occurs in real-life systems. The second innovation is to consider that finished products must be assembled before selling them. Indeed, to the best of our knowledge, this two-level lot-sizing and scheduling of processes that require auxiliary equipment is new. The last innovation is related to the batch production of the pieces which has not been considered in Ibarra-Rojas *et al.* (2011) nor in Ibarra-Rojas *et al.* (2010).

We use the structure of our proposed mathematical model for the PPMM problem to obtain a Relax&Fix heuristic to solve it. Indeed, this type of heuristic has already yield interesting results in the literature (Dillenberger *et al.*, 1994; Beraldi *et al.*, 2008; Escudero y Salmeron, 2005). In our work it will consist of making continuous relaxation of the variables that set the size of the final products and the number of pieces to be made. Thus, only the mold-machine assignment variables are left as integer and their obtained values are considered as fixed in the original model. This heuristic is fast and accurate but it cannot tackle larger and more realistic instances of the PPMM problem. Thus, we develop a multistart greedy algorithm (MGA) for obtaining feasible high quality solutions.

In Section 2.3 we present a new integer linear programming for the PPMM problem. Then, in Section 2.4, we develop a Relax&Fix heuristic for the PPMM problem while in Section 2.5 we solve the PPMM problem with the multistart greedy algorithm. In Section 2.6 we show that our iterative procedure for solving the PPMM problem is indeed efficient since the solutions obtained by the MGA are of high quality and have feasible mold scheduling solutions. Finally, in Section 2.7 we conclude and point out further areas of research for this problem.

2.3 INTEGER LINEAR PROGRAMMING FORMULATION

In this section we formulate the PPMM problem as an integer linear programming model. This mathematical structure will be helpful for designing the Relax&Fix heuristic of Section 2.4 and the MGA that is presented in Section 2.5.

First, let us suppose that a mold is installed twice on the same machine. Then, a solution that reduces the mold-setup times is the one that merges these two occurrences into one. This grouping idea is consistent with the way that companies make their production and has been reported to yield positive results (Chen y Wu, 2006).

Let $Finish$ be the set of the finished products, $Pieces$ the set of all pieces, $Molds$ the set of all available molds, and finally $Machs$ the set of all parallel machines. Each finished product f has a profit of c^f per item and a demand of d^f. Finished product $f \in Finish$ is composed by a set $Pieces(f)$ of pieces. A finished product may need more that one piece type to be assembled. Therefore, n_p^f corresponds to the number of pieces p needed to assemble product f (e.g. 2 pieces "eye" for a finished product "doll"). Each piece $p \in Pieces$ can be produced with molds of type $Molds(p)$. For manufacturing a piece p, a mold $m \in Molds(p)$ has cav_{pm} cavities. Notice that a given piece can be produced by different molds, then $|Molds(p)|$ is usually more than one. A mold m can be installed on a set of machines $Machs(m)$. Each machine k has an available time for production equal to $machTime_k$. Manufacturing a batch of piece p produced with mold $m \in Molds(p)$ that is installed on machine $k \in Machs(m)$ takes $batchTime_{pmk}$ time. The parameters it_{mk} represent the installation and removal time of mold m on machine $k \in Machs(m)$. The small preparation time between the execution of two different pieces on a single mold (color or a plastic change) are included in the processing time of the pieces.

Main decisions are about the amount to produce of each product f, $lotSize^f$. From these variables we can deduce auxiliary variables X_p^f that correspond to the number of pieces p manufactured to assemble product f. These two types of decisions are related to one another but notice that the price only depends on the number $lotSize^f$ of finished product f that is manufactured.

To formulate an ILP for the PPMM problem we also need three other sets of variables. Let variable $time_{pmk}$ correspond to the amount of time dedicated by machine k to produce piece p with mold m. With integer variable $numBatch_{pmk}$ we count the number of batches made by mold m to produce piece p on machine k. Binary variable Y_{mk} is equal to 1 if mold m is installed on machine k, and 0 otherwise. Additionally, let M be a very large number that will force variables Y_{mk} to be 1 when mold m is mounted on machine k. The value of M can be bounded by

$$\frac{machTime_k - it_{mk}}{batchTime_{pmk}}$$

for each piece p, mold m, and machine k. We can now state the ILP as follows:

$$\max \sum_{f \in Finish} c^f lotSize^f \tag{2.1}$$

$$\text{s.t. } lotSize^f \leq d^f \qquad f \in Finish \tag{2.2}$$

$$lotSize^f \leq \min_{p \in Pieces(f)} \left\lfloor \frac{X_p^f}{n_p^f} \right\rfloor \qquad f \in Finish \tag{2.3}$$

$$numBatch_{pmk} = \left\lfloor \frac{time_{pmk}}{batchTime_{pmk}} \right\rfloor \qquad p \in Pieces, m \in Molds(p), k \in Machs(m) \tag{2.4}$$

$$numBatch_{pmk} \leq MY_{mk} \qquad p \in Pieces, m \in Molds(p), k \in Machs(m) \tag{2.5}$$

$$\sum_{f \in Finish} X_p^f = \sum_{m \in Molds(p)} cav_{pm} \sum_{k \in Machs(m)} numBatch_{pmk} \qquad p \in Pieces \tag{2.6}$$

$$\sum_{m \in Molds} \left(it_{mk} Y_{mk} + \sum_{p \in Pieces} time_{pmk} \right) \leq machTime_k \qquad k \in Machs \tag{2.7}$$

$$lotSize^f, X_p^f, numBatch_{pmk} \in \mathbb{Z}^+ \qquad f \in Finish, p \in Pieces(f), m \in Molds(p), k \in Machs(m)$$

$$time_{pmk} \in \mathbb{R}^+, Y_{mk} \in \{0,1\} \qquad f \in Finish, p \in Pieces(f), m \in Molds(p), k \in Machs(m)$$

Objective function (2.1) maximizes the profit of selling the finished products. Notice that the aim is not to satisfy the demand as it is rarely fulfilled. Therefore, constraints (2.2) related to the demand are not often active. The limiting constraints are (2.3) that bound the number of finished products by the minimum number of pieces available to assemble complete products. Constraints (2.4) determine the number of batches $numBatch_{pmk}$ that correspond to the total time spent by mold m on machine k to produce piece p divided by the time required to produce a batch of piece p. Constraints (2.5) require Y_{mk} to be 1 when mold m is mounted on machine k, this happens whenever a batch of any piece p is produced with that machine and that mold assignment. Indeed, the maximum number of batches of piece p that could be made with mold m on machine k would correspond to the situation where the only mold installed on machine k is m during all the time horizon. Constraints (2.6) compute the total amount of pieces p as the number of batches of the molds that produce this piece times the number of cavities of the molds. Each machine k can only be used $machTime_k$ of time. This time is consumed by the installation times of the molds on this machine plus the amount of time used by the molds to produce the pieces and this is reflected in constraints (2.7).

A solution of the PPMM problem indicates the mold-machine assignments together with the time that a mold is used on each machine (including its setup time). In the scheduling terminology, one would say that the jobs (corresponding to the time that molds are used on each machine) are assigned to the machines. Although the PPMM problem does not give the starting (or completion) times of the jobs, with the following property we enable a set of solutions of the PPMM problem that will have an unfeasible scheduling. In particular, for any mold $m \in Molds$, an upper bound on the maximum time that this mold can be used (setups plus execution times) is equal to the maximum available machine time.

PROPERTY 2.1. *A solution of the PPMM problem that has a feasible mold scheduling on the machines satisfies the following sufficient condition:*

$$\sum_{k \in Machs} \left(it_{mk} Y_{mk} + \sum_{p \in Pieces} time_{pmk} \right) \leq \max_{k} machTime_k \quad m \in Molds. \tag{2.8}$$

By Property 2.1, we can add constraints (2.8) to the ILP of the PPMM problem to reduce the number of solutions that will have an unfeasible mold scheduling on the machines, that is, solutions that have mold overlapping. This is the key point of our study since we are not including all the variables and constraints that would enable solutions without mold overlapping. By relaxing the problem we can obtain solutions that are most of the times free of mold overlapping and *easier* to obtain. Nevertheless, the PPMM problem is hard to solve as stated by the following theorem.

THEOREM 2.2. *The PPMM problem belongs to the NP-hard complexity class.*

Proof. The decision version of this problem is clearly in NP. The rest of the proof is by restriction since the structure of constraints (2.7) is similar the one of the Multiple Knapsack problem (Martello y Toth, 1990; Chekuri y Khanna, 2005) that is NP-hard. Consider an instance of the PPMM problem such that one mold can make only one type of piece and each product is made by only one piece, then we have an instance that is identical to the Multiple Knapsack problem. □

The alternative of introducing in the PPMM model the constraints that guarantee that there would be no mold overlapping, that is, that the molds have a feasible scheduling on the parallel machines, implies a quadratically constrained integer model that has no convex continuous relaxation. These quadratic models are usually harder to solve. For our case study, some preliminary experimental results yield non-global solutions of poor quality when solving the quadratic model even when applying convexification reformulation methods as the ones of Plateau y Rios-Solis (2010).

In Section 2.6 we show that solving the PPMM's problem ILP by an integer linear solver is not efficient for real size instances. Therefore, we propose a Relax&Fix heuristic in Section 2.4 and a greedy randomized adaptive search procedure in Section 2.5 that are based on the ILP model of the PPMM problem.

2.4 RELAX&FIX HEURISTIC

The integration of lot sizing and scheduling decisions generally results in difficult problems. Even if we only consider a single period in this study, the structure of ILP developed for the PPMM problem suggests that a simple but effective Relax& Fix heuristic could be used (Beraldi *et al.*, 2008; Escudero y Salmeron, 2005).

In this procedure, we make a continuous relaxation of the variables that set the size of the final products, $lotSize^f$, the number of pieces to be made, X_p^f, and the number of batch production cycles $numBatch_{pmk}$. Thus, only the mold-machine assignment variables Y_{mk} are left as integer. Let ILP' be the resulting relaxed model. After solving ILP' with a branch-and-bound algorithm up to optimality, we have the values of the continuous relaxed variables $numBatch'_{pmk}$. Then, the Relax&Fix heuristic solves the original ILP but this time, variables $numBatch_{pmk}$ are fixed to $\lfloor numBatch'_{pmk} \rfloor$.

Different combinations of this Relax&Fix procedure were tested but only this one yielded interesting results that will be shown in Section 2.6. Nevertheless, large instances could not be solved by this Relax&Fix heuristic so in the next section we present a multi-start greedy algorithm to solve them.

2.5 MGA: MULTISTART GREEDY ALGORITHM FOR THE PPMM PROBLEM

To solve the PPMM problem, we propose a multistart greedy algorithm procedure, MGA (Cormen, 2009; Resende y Ribeiro, 2014). In the structure of the PPMM model we can notice that for the same amount of finished products, there can be many different piece-mold-machine assignments. Moreover, we did some preliminary experimentation with classical scheduling local searches (with insertion, removal, swap operators applied to the set of pieces and/or machines) and it did not yield high quality solutions in reasonable time. Therefore, our MGA focuses on a local search defined on the finished products, that is, on variables $lotSize^f$ rather than on the pieces or the mold-machine assignments.

The procedure of the MGA for the PPMM problem is described in Algorithm 1. Each MGA iteration consists of two phases, a construction phase based on a greedy function in which a solution is produced, and a local search phase, in which a local optimum in the neighborhood of the constructed solution is sought.

Before presenting the MGA, we need to specify the information contained in a solution s of the PPMM problem:

- The amount of finished products: $\boldsymbol{lotSize} = \{lotSize^f | f \in Finish\}$.
- The number of pieces that are made for each finished product: $\boldsymbol{X} = \{X_p^f | f \in Finish, p \in Pieces(f)\}$.
- The time that each mold spends on each machine to produce a certain piece: $\boldsymbol{time} = \{time_{pmk} | p \in Pieces, m \in Molds(p), k \in Machs(m)\}$. From these values we can compute the remaining time that a machine or a mold has left for production.
- The values of the binary variables $\boldsymbol{Y} = \{Y_{mk} | m \in Molds, k \in Machs(m)\}$ that are equal to 1 if mold m is installed on machine k, and 0 otherwise.

Summarizing, $s = (\boldsymbol{lotSize}, \boldsymbol{X}, \boldsymbol{time}, \boldsymbol{Y})$.

The MGA starts with the trivial solution $s^* = (\boldsymbol{lotSize} = 0, \boldsymbol{X} = 0, \boldsymbol{time} = 0, \boldsymbol{Y} = 0)$ and with ordered list $UnFinish$ consisting of the products f that have not yet been processed. Each product has the following probability to be drawn from the list:

$$\frac{c^f}{\left(\sum_{p \in Pieces(f)} n_p^f\right)\left(\sum_{f' \in UnFinish} \sum_{p \in Pieces(f')} n_p^{f'}\right)}, \tag{2.9}$$

which takes into account the price of the finished product with respect to the number of pieces that it needs to be assembled. Notice that the second term of the denominator is a normalization with respect to the other products in $UnFinish$, in order to have a probability. In this manner, finished products that yield a higher benefit and that need fewer pieces

Algorithm 1 MGA: Multistart Greedy Algorithm for the PPMM problem

```
1:  $s^* = (\boldsymbol{lotSize} = 0, \boldsymbol{X} = 0, \boldsymbol{time} = 0, \boldsymbol{Y} = 0)$, populate list $UnFinish$
2:  while stop criterion not reached do
3:      $s = (\boldsymbol{lotSize} = 0, \boldsymbol{X} = 0, \boldsymbol{time} = 0, \boldsymbol{Y} = 0)$
4:      while $UnFinish \neq \emptyset$ do
5:          $f^*$ drawn from $UnFinish$, $p^*$ drawn from ordered set $Pieces(f^*)$
6:          $lotsize^{f^*} = \min\left\{d^{f^*}, UB^{f^*}_{p^*}\right\}$ is the upper bound of the product $f^*$ to be produced
7:          if $lotsize^{f^*} = 0$ then
8:              $UnFinish = UnFinish \setminus \{f^*\}$
9:          else
10:             $\Delta = 1, Flag = 0$
11:             while $Pieces(f^*) \neq \emptyset$ or $Flag \neq 1$ do
12:                 compute $\mathscr{X}^{f^*}_{p^*k} = \max_{\substack{m \in Mold(\overline{p}) \\ k \in Mach(m)}} \left\lfloor \frac{cav_{\overline{p}mk}(\min(machTime_k, moldTime_m) - it_{mk}(1-Y_{mk}))}{n^{f^*}_{\overline{p}} \cdot batchTime_{\overline{p}mk}} \right\rfloor \forall k$
13:                 if $\mathscr{X}^{f^*}_{p^*k} = 0$ for all $k$ then
14:                     if $lotsize^{f^*}\Delta \leq 1$ then $Flag = 1$, remove $f^*$ from $UnFinish$ , erase $f^*$
                        from $s$
15:                     else reduce $\Delta$, restore $s$
16:                 else
17:                     $\mathscr{X}^{f^*}_{p^*} = \min\{\max_k \mathscr{X}^{f^*}_{p^*k}, lotsize^{f^*} n^{f^*}_p - X^{f^*}_p\}$
18:                     update in $s$: $X^{f^*}_p = X^{f^*}_p + \mathscr{X}^{f^*}_{p^*}$, $\boldsymbol{time}$, and $\boldsymbol{Y}$
19:                     if $lotsize^{f^*}\Delta n^{f^*}_{p^*} - X^{f^*}_{p^*} = 0$ then remove $p^*$ from $Pieces(f^*)$
20:                 end if
21:             end while
22:         end if
23:     end while
24:     if $s$ has a better objective function that $s^*$ then $s^* = s$
25: end while
```

to be assembled have the highest probability to be processed. Other criteria were tested but they did not show as good performance in the quality of the final solution than this simpler one.

Each iteration performs the following procedure (see Algorithm 1). In 5 we draw a finished product f^* from $UnFinish$ with respect to the probability previously established by (2.9). For f^* we order set $Pieces(f^*)$ in no-ascending order with respect to the following equation that determines the mold-machine assignment that would produce more pieces p without taking into account other mold-machines assignments:

$$UB_p^{f^*} = \max_{\substack{m \in Mold(p) \\ k \in Mach(m)}} \left\lfloor \frac{cav_{pmk}(\min(machTime_k, moldTime_m) - it_{mk}(1 - Y_{mk}))}{n_p^{f^*} \cdot batchTime_{pmk}} \right\rfloor . \quad (2.10)$$

Thus, the most constraining piece p^* (the bottleneck piece) is the first in $Pieces(f^*)$. This bottleneck piece is the one that will bound the lot size of f^* which corresponds to verifying constraints (2.2) and (2.3) of the ILP. If this bound is equal to zero then finished product f^* cannot be manufactured and it is removed from $UnFinish$ (step 8). Otherwise, we start a simple local search algorithm to find the amount of product $lotsize^{f^*}$ that makes s a feasible solution (steps 10-19). This local search initializes with $\Delta = 1$ so the local search will try to arrange the pieces of f^* in order to make $lotsize^{f^*}$. If it fails, that is, there is no feasible solutions that can be found, then Δ will be reduced thus the algorithm will make less than $lotsize^{f^*}$. In step 11, we determine the amount of piece p^* that could be made on each one of the machines. If there is no machine that can handle p^* we have two cases. The first one is that we have reduced so much the lot size that it is less than one. In this case we remove the product f^* from $UnFinish$, remove also all information of f^* in s and search for another product. Else, we have to reduce the amount of lot size of f^* by reducing Δ, and we delete from s all the information related to f^* and give p^* a higher priority in $Pieces(f^*)$. In the case where there is at least one machine where piece p^* can be produced (step 17), we choose the mold-machine pair that can produce more of it and we update this assignment in s. Notice that a piece p^* can have several mold-machine assignments in s. If all the pieces p^* to produce f^* have been made, we remove p^* from $Pieces(f^*)$. Finally, in step 24 we update the best solution s^* if the current solution s has a better objective function value.

With respect to the local search, given s, a neighbor of this solution is obtained by decreasing its correspondent value of $lotsize^{f^*}$ by the proportion Δ. The neighborhood $Neigh(s)$ of s is composed by all of its neighbors. Notice that any neighbors of s (even itself) may not have feasible piece-mold-machine assignments since the execution time of the pieces that make the fixed number of final products may be larger than the available time of the machines. Therefore, this simple local search will generate a suite of neighbors until it finds a feasible one.

THEOREM 2.3. *The MGA for the PPMM problem has a complexity of* $O(\max_i d_i |Finish||Molds||Machs||Pieces|^2)$.

Proof. The more time consuming step is the while loop of the local search (step11). Indeed, for each finished product and for each piece of the product, step 15 can be done, in the worst case, until the demand of the finished product is 0 (Δ would reduce $lotsize^{f^*}$ one by one). Moreover, restating the current solution in this step takes $O(|Pieces||Molds||Machs|)$ which gives the stated complexity of the PPMM's MGA. □

With the MGA for the PPMM problem we have obtained a solution where the size of the finished products to be manufactured is set together with the piece-mold-machine assignments. Nevertheless, this solution may not have a feasible scheduling since two molds could be used by two different machines at the same time. We must check that this solution is feasible (we could use the feasibility scheduling algorithm of Ibarra-Rojas *et al.* (2011)). If the solution is not feasible, we run the MGA again but restrict the mold-machine assignments of the current solution to obtain a new one. This procedure could in theory be exponential but in practice the MGA obtains in most cases a feasible solution for the PPMM problem in one iteration as it will be shown in the next section.

2.6 EXPERIMENTAL RESULTS

In this section we empirically evaluate the efficiency of the ILP, the Relax&Fix heuristic, and the MGA to solve the PPMM problem.

We generated a set of 150 instances based on observed data from a real plastic injection company. The size of an instance is determined by the number of finished products, pieces, molds, and machines. We denote the specific instance by means of a 4-tuple: $(|Finish|, |Pieces|, |Molds|, |Machs|)$. The difficulty of an instance not only resides on its size but also on the number of finished products that need a specific piece, the number of molds that can make each piece, and the number of machines where a mold can be installed. We name these *compatibility factors* as: $\delta_{Finish-Pieces}$, $\delta_{Pieces-Molds}$, and $\delta_{Molds-Machs}$, respectively. More precisely, a compatibility of $\delta_{Pieces-Molds}$=25% means that each piece has a probability of being made by at most 25% of the molds. In Ibarra-Rojas *et al.* (2011) the authors show that the difficulty of an instance does not reside on the $\delta_{Pieces-Molds}$ and $\delta_{Molds-Machs}$ compatibility factors, so we fix them close to the real observed compatibility, that is, $\delta_{Pieces-Molds} = 15\%$ and $\delta_{Molds-Machs} = 60\%$. Nevertheless, in this work we show that the Finish-Piece compatibility factor is a vital component in determining the difficulty of an instance. Therefore, we establish three different compatibility factors for $\delta_{Finish-Pieces}$: 25%, 50%, and 100%. The observed compatibility factor in the company between Finish and Pieces is around 25%. A $\delta_{Finish-Pieces} = 100\%$ would mean that all pieces are needed by all the finished products but probably in different quantities. Note that the instances from Ibarra-Rojas *et al.* (2011) do not consider finished products so we cannot compare our algorithms with theirs.

In Figure 2.3 is illustrated a *small* instance with 5 finished products ($|Finish| = 5$), 50 pieces ($|Pieces| = 50$), 30 molds ($|Molds| = 30$), and 5 machines ($|Machs| = 5$). The compatibility between the different sets for this instance is as follows: $\delta_{Finish-Pieces} = 25\%$, $\delta_{Pieces-Molds} = 15\%$, and $\delta_{Mold-Machs} = 60\%$.

The instances are classified with respect to their size into *T1* (5,50,30,5), *T2*

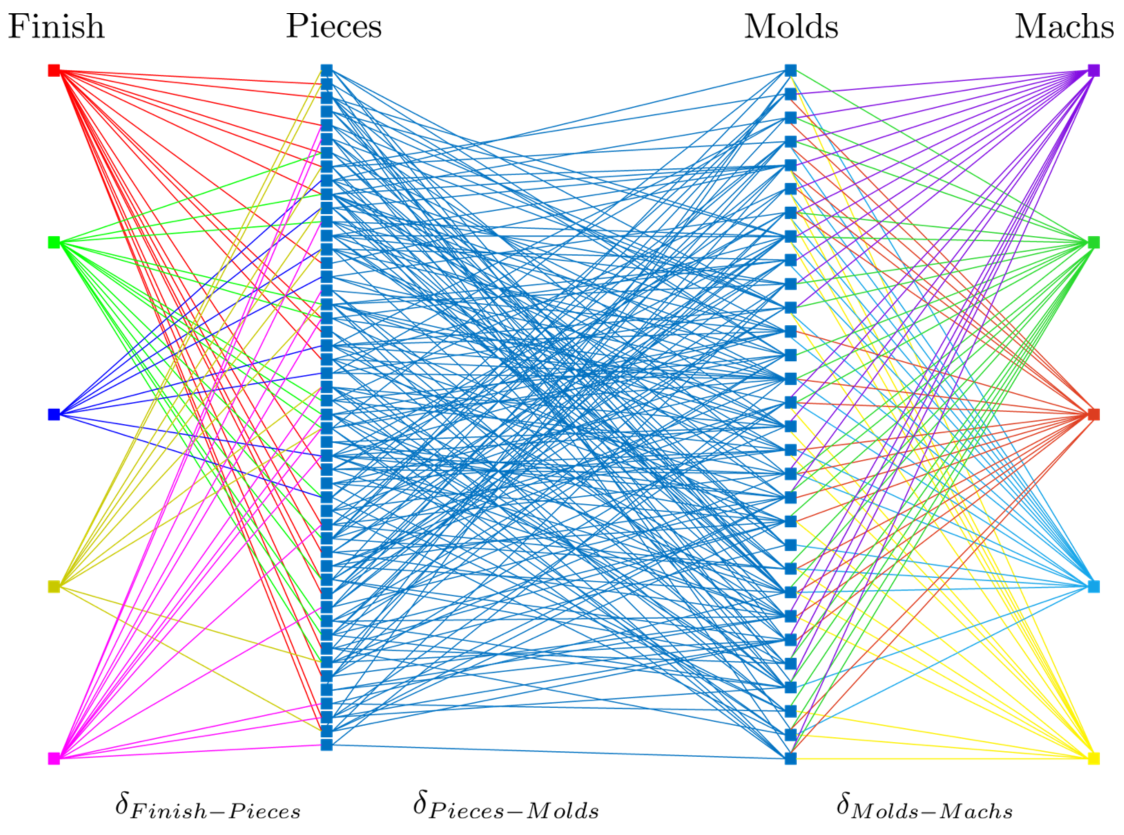

Figure 2.3: Instance of size (5,50,30,5): 5 types of finished products, 50 pieces that conform the finished products, 30 molds to make the pieces and 5 machines to install the molds. The compatibility factors are $\delta_{Finish-Pieces} = 25\%$, $\delta_{Pieces-Molds} = 15\%$, and $\delta_{Mold-Machs} = 60\%$

(10,120,80,20), *T3* (20,200,120,25), *T4* (30,350,200,50), and *T5* (50,450,250,75). Instances *T3* and *T4*, correspond to the observed setting faced by the company we have been collaborating with. For each size we randomly generate 30 different instances. The first 10 instances of each size have a compatibility factor $\delta_{Finish-Pieces}$ of 25%, the second ones of 50%, and last ones of 100%. Our instance generator was coded in C++ and guarantees that for all instances, each finished product has at least one piece, each piece must belong to at least one finished product, each piece can be produced by at least one mold, each mold produces at least one piece, each mold can be mounted on at least one machine, and finally, on each machine at least one mold can be installed. All the instances can be found in the Online Resource.

For the rest of the instance parameters we used the observed values of the company we are basing our study on. The number of pieces p needed by finished product f, n_p^f, is randomly selected between 1, 2 or 3 pieces. The demand of the finished products $d^f \in [1000, 18000]$. The mold-setup times needed for installing and uninstalling a mold m on machine k, it_{mk}, are between 90 and 138 minutes. Indeed, recall that the molds must be moved by cranes. The number of cavities of mold m for piece p, $cav_{pm} \in [4, 17]$. The time needed for a batch of a certain piece depends on the type of machine, mold, and material used (plastic in our case of study). We set the time taken for producing a batch of piece p with mold m on machine k, $batchTime_{pmk}$ as either 1 or 2 minutes. All machines work 24 hours a day, therefore the time available for each machine k is $machTime_k = 1440$ minutes. This research does not take into account maintenance times or plant shutdowns. Finally, the price of a finished product c^f corresponds to the sum of the estimated value of its pieces that is between $[5, 10]$[1].

We will compare three algorithms that were executed on a PC with Intel Core i7-4790K Processor, 8M Cache, up to 4.40 GHz, and 8 GB of RAM:

B&B: CPLEX 12.6 linear solver. We used the default settings except for the optimality gap which we set to 0. B&B(1h) indicates that the stop criterion is of one hour while B&B(5m) indicates that it is of 5 minutes.

Relax&Fix: heuristic proposed in Section 2.4 implemented in CPLEX 12.6. We also used the default settings except for the optimality gap which we set to 0. Relax&Fix(1h) has a stop criterion of one hour for ILP' (the relaxed model) and ILP (the one with the fixed variables) while Relax&Fix(5m) has it of 5 minutes for both ILP' and ILP.

MGA: multistart greedy algorithm described in Section 2.5, coded in C++. The stopping criterion for the MGA is when there is no improvement of the value of the solution

[1] Another set of instances where the number of pieces needed by finished product is between $[1, 5]$ and where the price of the final products is independent from the number of pieces, has been tested and is commented later.

after 20 iterations. The acceptance criterion is to have a solution with a better value of the objective function. After preliminary experimentation, we have set $\Delta = 0.99$ reduce at each iteration the lot-size of the chosen product.

Table 2.1 shows the results of solving the PPMM problem with algorithms B&B, Relax&Fix, and MGA. Each row is the average of 10 different instances. In the first column we state the size of the instance. For example, *T1*-25 corresponds to an instance of the set *T1* with $\delta_{Finish-Pieces} = 25\%$.The second and third columns correspond to the average Gap calculated as: (best integer solution - best dual relaxation)/ best integer solution, and the average time in seconds for the B&B(1h) algorithm. The rest of the columns are analogous than the previous ones but for Relax&Fix(1h), B&B(5m), Relax&Fix(5m), and the MGA. The Gap for the MGA is computed as: (solution MGA - best dual relaxation of B&B or Relax&Fix)/best dual relaxation of B&B or Relax&Fix. For the *T4* and *T5* sets, neither the B&B nor the Relax&Fix algorithms were able to solve any of the instances ("–" in Table 2.1), they could not even compute the first linear relaxation of the B&B. Therefore, in the MGA we could not compute the Gap for these instances. Notice that CPLEX checks before each call of the optimizer if the time limit is reached, hence sometimes the time limit is exceeded by some seconds in our experiments.

In Table 2.1 we can observe that only for the *T1* instances, the B&B(1h) yield the best results with respect to the Gap. Then, for *T2* and *T3* it is the Relax&Fix(1h) procedure that gives the best results for the Gap. For all of the instances the best results with respect to the time correspond of course to the MGA which is the only procedure that solves the *T4* and *T5* instances. When the compatibility factor $\delta_{Finish-Pieces}$ is equal to 100% it implies that all the finished products need all the pieces (the number of each type piece needed may differ). Therefore, we have a complete Finish-Pieces subgraph leading to a high number of decision variables. It is interesting to notice that the instances with $\delta_{Finish-Pieces} = 100\%$ are not always the hardest instances for the exact procedures B&B and Relax&Fix. Indeed, some symmetries may arise in these cases and are easily excluded by the branch-and-bound procedures. An important aspect of the MGA is that the quality of the solutions is maintained even if we increase the size of the instances. The MGA

Table 2.1: Results of solving the generated instances with algorithms B&B(1h), Relax&Fix(1h), B&B(5m), Relax&Fix(5m), and the MGA

	B&B(1h)		Relax&Fix(1h)		B&B(5m)		Relax&Fix(5m)		MGA	
Inst.	Gap	Time	Gap	Time	Gap	Time	Gap	Time	Gap	Time
T1-25	**0.01**	2041.3	0.76	2.1	0.03	242.9	0.76	2.1	17.94	**0.05**
T1-50	**0.08**	2784.2	0.43	1.8	0.12	241.3	0.43	1.8	20.64	**0.05**
T1-100	**0.17**	3329.4	0.51	4	0.21	300.4	0.51	4	23.71	**0.07**
T2-25	4.47	3670.7	**3.53**	3614.6	6.71	300.6	3.96	301.8	21.34	**0.72**
T2-50	4.52	3651.8	**3.08**	3615.3	14.61	300.8	4.35	301.9	21.92	**0.99**
T2-100	4.04	3686.5	**2.29**	3659.6	6.51	300.7	4.1	302	19.21	**1.42**
T3-25	9.16	3601.5	**5.31**	3608.8	46.61	302	14.81	310.6	26.64	**2.06**
T3-50	7.36	3601.3	**4.93**	3608	53.19	302.1	44.93	337.7	26.63	**3.07**
T3-100	10.31	3734.6	**5.87**	3678.7	19.17	302.2	7.60	313.7	24.83	**4.74**
T4-25	–	–	–	–	–	–	–	–	–	15.84
T4-50	–	–	–	–	–	–	–	–	–	24.09
T4-100	–	–	–	–	–	–	–	–	–	41.13
T5-25	–	–	–	–	–	–	–	–	–	53.03
T5-50	–	–	–	–	–	–	–	–	–	90.86
T5-100	–	–	–	–	–	–	–	–	–	156.72

rapidly converges to a local optimum that is not easy to escape from even with a drastic random perturbation. So, giving more execution time to the MGA does not increases its quality. Notice that there are some set of instances where the Gap of the MGA is better than the Relax&Fix(5m) and the B&B(5min) like the *T3*-50 of *T3*-25 instances.

We executed another set of instances where the number of pieces needed by finished product is between $[1,5]$. The aim was first to determine if this was a key component of the difficulty of the instance but the experimental results showed that this was not a sensible parameter. The fact that the prices of the final products are independent from the number of pieces gives rise to slightly easy to solve instances since all the algorithms easily detect the products with higher price and less pieces and favor their production.

A major insight into the problem obtained from our research is the fact that most of the solutions obtained by our methodologies do not need the extra constraints to avoid a mold overlapping. As mentioned before, once the B&B , Relax&Fix or the MGA obtain a solution, we must check if this solution has a feasible scheduling of the molds. In 99% of the instances, the solutions have a feasible scheduling of the molds on the machines, that is, there is no mold overlapping. Only one instance of *T2* with $\delta_{Finish-Pieces} = 25\%$ did not have a feasible scheduling of the molds when solved with the MGA. Therefore, we executed the MGA a second time, but this time we prohibited the mold that showed more overlapping to be installed in more that two machines. This time the MGA solution had a feasible scheduling with a solution of equal value of the objective function.

2.6.1 CALIBRATION OF THE MGA

There are two main parameters in the MGA: the number of iterations without improvement (stop criterion), and the Δ parameter that reduces the amount of product that can be made at each iteration (step 15 of Algorithm 1). Regarding the number of iterations, we fixed it to 20 since most of the instances behave as the ones presented in Figure 2.4 where the compatibility factor is $\delta_{Finish-Pieces} = 100\%$. In the abscissa axis we have the number of iterations while in the ordinate one we present the Gap (using the best dual solution of the B&B). Notice that the inflection point for all the curves has already appeared before the 20-th iteration.

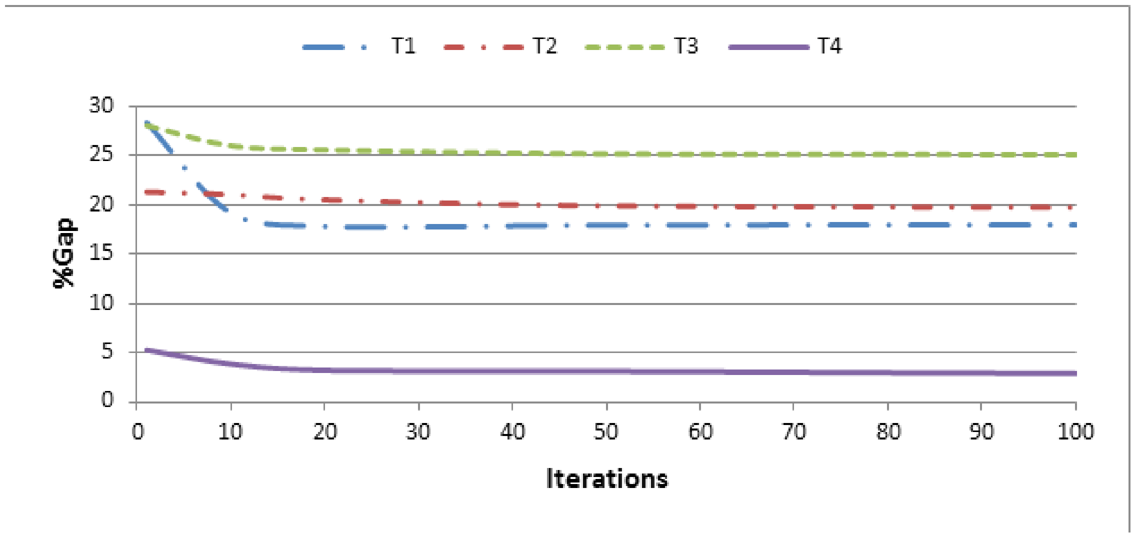

Figure 2.4: Gap versus iterations for the MGA algorithm for *T1*-*T4* instances with $\delta_{Finish-Pieces} = 100\%$

A similar analysis was made for Δ. Our experiments showed that $\Delta = 0.99$ generates the best solutions without requiring too much computational resources. On the one hand, when Δ is smaller than 0.99, then the MGA can miss some high quality solutions. On the other hand, the number of iterations that the algorithm performs when $\Delta = 0.99$ does not drastically increase the execution times.

An obvious question is: What happens if the B&B is given more time? In Figure 2.5 we show the Gap versus time in hours for one instance of each set *T1*-*T3* with $\delta_{Finish-Pieces} = 100\%$ (Gap % in the Y axis and hours in the X one). We can notice that the inflection point is between one and four hours for most of the instances.

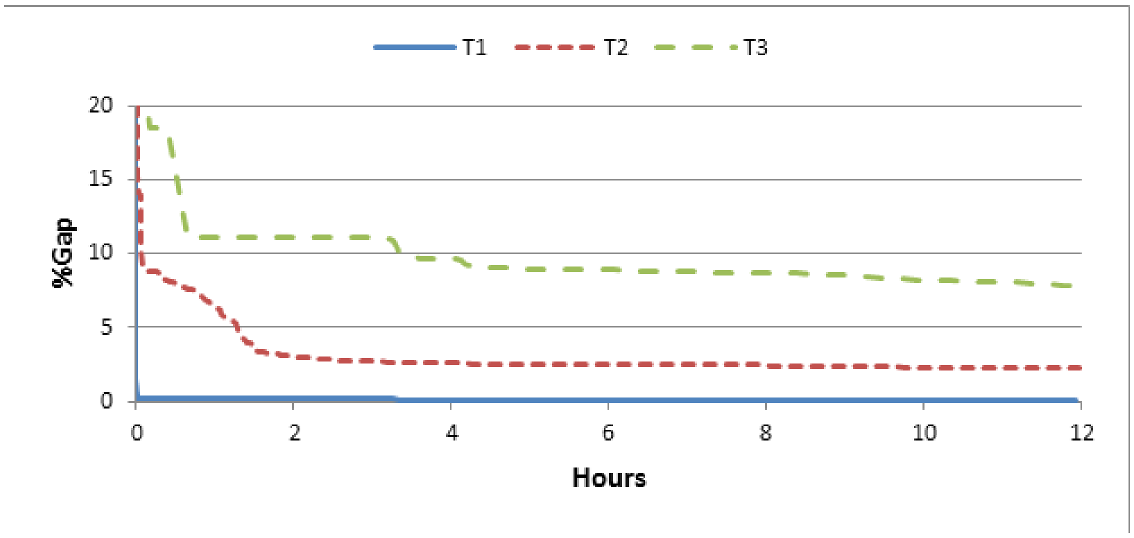

Figure 2.5: Gap versus time in hours for the B&B algorithm for the *T1*-*T3* instances with $\delta_{Finish-Pieces} = 100\%$

2.7 CONCLUSIONS

The Product-Pieces-Mold-Machine problem is about the manufacturing of a set of pieces that assemble finished products, each piece has a set of molds that can be used to make it, molds can be assigned to a certain number of machines and, setup times arise each time a new mold is installed on a machine. The aim of this problem is to determine the single period lot production size of each finished product, the piece-mold and the mold-machine assignments such to maximize the benefits of selling the finished products. The key point of our method is that we have dropped the constraints that guarantee a feasible scheduling of the molds on the machines. Without these constraints, our mathematical model becomes linear and from its structure we can derive a a Relax&Fix heuristic and a multistart greedy algorithm.

Experimental results on generated instances show the efficiency of our approach for different sizes of the instances of the problem and validate that 99% of the instances verify the mold scheduling constraints in a natural way.

There are obvious research lines to pursue. The first one is the consideration of several periods as most of the lot-sizing problems deal with. The structure of the multiperiod PPMM problem is different from the one proposed here so new models and methods should be developed. Another future research leads to include batching sizes of the production to consider a more efficient way of assembly of the finished products. Indeed, in a manufacturing enterprise, the assemble line must reduce its idle times. Another research line deals with the fact that the more time a mold is used the least the quality of the pieces that it is producing. Here, questions about the ideal time for replacing the molds arise and make the problem have several objective functions.

The PPMM problem and the methodology we have proposed here has already been tested in some plastic injection enterprises. Its value has resided in the minimization of the time the planner had to use to make a production plan and in the ability to make strategic decisions and corroborate them with this model: What happens when one or more machines are added? What if we buy an extension to a mold so it can be used in other machines?

CHAPTER 3

BUS HOLDING FOR A SIMULATED TRAFFIC NETWORK

This research produced the following book chapter:

> Hernández-Landa, L. G., Morales-Marroquín, M. L., Nigenda, R. S., Ríos-Solís, Y. Á. (2015). Linear Bus Holding Model for Real-Time Traffic Network Control. In Applied Simulation and Optimization (pp. 303-319). Springer International Publishing.

3.1 ABSTRACT

One of the most annoying problems in urban bus operations is *bus bunching*, which happens when two or more buses arrive at a stop nose to tail. Bus bunching reflects an unreliable service that affects transit operations by increasing passenger-waiting times. This work proposes a linear mathematical programming model that establishes bus holding times at certain stops along a transit corridor to avoid bus bunching. Our approach needs real-time input, so we simulate a transit corridor and apply our mathematical model to the data generated. Thus, the inherent variability of a transit system is considered by the simulation, while the optimization model takes into account the key variables and constraints

of the bus operation. Our methodology reduces overall passenger-waiting times efficiently given our linear programming model, with the characteristic of applying control intervals just every 5 minutes.

3.2 INTRODUCTION AND PROBLEM DESCRIPTION

The study of complex bus operating systems is usually divided in two main areas, *line planning* and *real-time control* (Ceder, 2007; Desaulniers y Hickman, 2007). The *line planning* process involves strategic, tactical, and operational decisions. Strategic problems relate to long-term network design decisions. Tactical and operational decisions ultimately define the service offered to the public; for example, frequency of buses, definition of stops, bus timetabling, vehicle scheduling, driver scheduling, maintenance scheduling, among other problems.

Real-time control tries to maintain the bus system operational along the day in order to minimize passenger inconvenience caused by the inherent stochastic dynamics of the network or traffic situations (Desaulniers y Hickman, 2007). Although, bus frequency is planned for each stop in the network, changes in the passenger flow, traffic, or even in the timetabling, produce perturbations that give rise to one of the most annoying problems in urban transportation operations, the *bus bunching problem* (BBP) that happens when two or more buses arrive at a stop nose to tail. BBP is one of the most common customer complaints in today's networks since it reflects an unreliable service that affects transit operations by increasing passenger-waiting times.

In Figure 3.1, we show the causes of bus bunching for a single bus line with three trips, that have the following timetable: 8:00, 8:15, and 8:30. For the four graphs, time is represented by the x-axis, while the first two stops are represented by the y-axis. The first graph shows how the planning should look like if everything were deterministic. We can see that the lines of the three trips are *parallel*, so the time differences between them (called headways) are of exactly 15 minutes. The second graph shows the perturbations

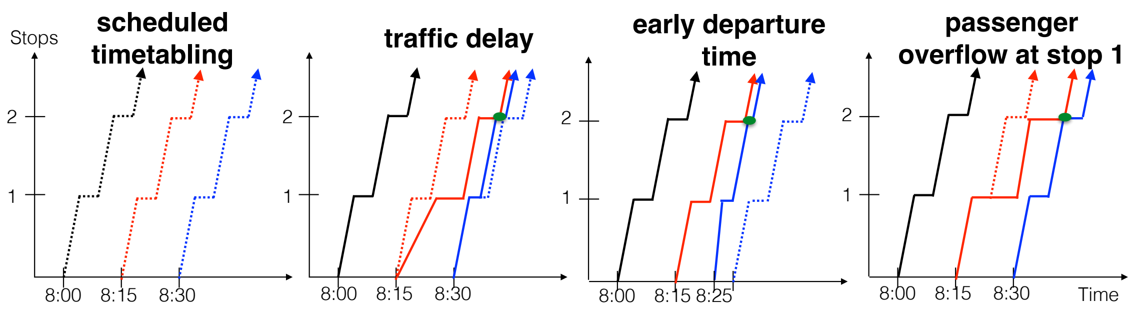

Figure 3.1: Causes of bus bunching (modified from Ceder (2007)).

that arise when a traffic delay hits the second trip between the depot and the first stop. The doted lines are the planned schedules, while the plain lines are the real executed delayed plans. Since the 8:15 bus takes longer to arrive to stop 1, then there are more passenger waiting to board it. When the bus that departed at 8:30 arrives at stop 1 many of the passengers that should have board it, already boarded the 8:15 bus. Then, these two buses will bunch close to stop 2. Graph three represents bunching situations when the departure time of a trip is moved earlier. Similarly to the second case, there will be less passengers at stop 1 so the bus will go faster and catch the 8:15 bus around stop 2. Finally, the fourth graph considers the case of passenger overflow. This graph shows that since there are extra passengers at stop 1 so the dwell time of the second bus at that stop will be longer. In other words, the second bus is taking passengers that would be normally assigned to the third bus. By the time the second and third bus arrives at stop 2, they are generating a bus bunching situation.

In this work, we provide solutions to the *bus bunching problem* by maintaining *congruent headways*. Furthermore, we will show that maintaining congruent headways implicitly reduces passenger-waiting times. As mentioned, the headway is a quality measure given to the time difference between two consecutive buses. A bus line could have equally distant headways or different ones for each pair of buses Ceder (2001); Ibarra-Rojas y Rios-Solis (2012); Ibarra-Rojas *et al.* (2014). We say that *headways are congruent* if the real-time differences between buses are nearly identical to the originally planned. Headway congruence does not necessarily comply with planned timetables. Indeed, the time when a bus arrives at a stop may not be the planned one, but if the distance to his predecessor is almost the planned headway, then it will be a congruent headway. Congruent headways reflect a reliable service especially for the cases when timetables are not intended for the public so the users only know estimated headways for the lines as in Monterrey, Mexico, and many Latino-American cities.

Our methodology interleaves optimization and real-time data retrieving to maintain congruent headways and solve BBP along the day. During the optimization phase, a linear programming model is built and solved to exactly determine the holding times of the buses at the stops in order to maintain congruent headways. The real-time data retrieving phase indicates, at every interval of time, the positions of the buses along a single corridor where only one line operates on a given frequency. In Figure 3.2, we can observe how optimization and real-time data retrieving interleave. Real-time (or simulated) data are acquired from the bus corridor to obtain the distance between each bus and its last visited stop, together with the number of passengers waiting at each stop. Then, these data are used to populate our linear programming model, which yields the optimal holding times for each bus in the corridor.

Most of the works in the literature base their quality measure on the waiting times of the passengers, or the variance between the departure times of the buses at the stops, which are generally modeled with quadratic functions that are harder to solve and therefore difficult to operate by real-time systems. By using a linear objective function that minimizes the penalties arising when headways are not congruent, our methodology returns optimal solutions in short time. One of the main contribution of this work is that by maintaining congruent headways, we implicitly reduce the overall passenger waiting and travel times, as our experimental results will demonstrate.

The rest of this chapter is structured as follows. A brief revision of the state of the art is presented in Section 3.3. In Section 3.4, we present our new linear programming model inspired in earliness and tardiness penalties of just-in-time scheduling problems, which determines the optimal holding times of the buses at the stops. Then, Section 3.5 shows the efficiency of our model on a discrete event simulation of a single corridor.

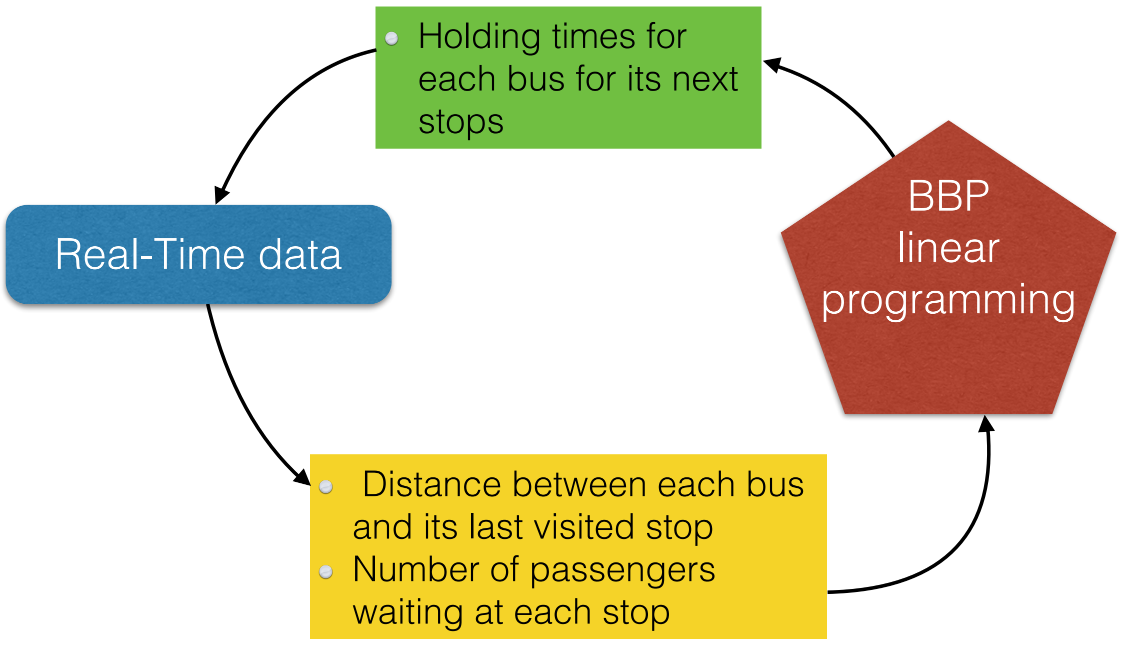
Holding times for each bus for its next stops
Real-Time data
BBP linear programming
Distance between each bus and its last visited stop
Number of passengers waiting at each stop

Figure 3.2: Framework for interleaving optimization (BBP LP modeling) and real-time data retrieving (or simulation).

Finally, Section 3.6 presents our conclusions, and discusses open research questions that arise from this work.

3.3 STATE OF THE ART RESEARCH IN REAL-TIME BUS OPERATIONS

Most of the literature related to real-time bus operations deals with models that have non-linear objective functions. Therefore, the holding times that each bus most be held at the stations are approximations. Work by Zhao *et al.* (2003) minimizes the average waiting cost of passengers, including both off-bus and on-bus costs that are non-linear, when there is no capacity imposed to the buses. Eberlein *et al.* (2001) minimize the variance between the departure times, which is a quadratic function, and therefore propose heuristic solutions. Sun y Hickman (2008) propose a convex quadratic programming problem to minimize the variance between the departure times. A closer work to ours is proposed by Ding y Chien (2001), since they consider the minimization of the total variance of headways between buses in all stops.

Daganzo (2009) and Daganzo y Pilachowski (2011) propose adaptive control schemes aiming to provide quasi-regular headways, while maintaining as high commercial speed as possible. In Daganzo y Pilachowski (2011) the authors continuously adjust bus cruising speed based on a cooperative two way based approach that considers the headways of the previous and posterior buses. Bartholdi III y Eisenstein (2012) abandon the idea of any *a priori* target headway, allowing headways to dynamically self-equalize by implementing a simple holding rule at a control point. It is worth noting that the aim of the previously mentioned studies is to maintain equally headways so they do not consider timetables where the headways may be different for each pair of buses and they are not apt for situations when the buses reach their capacities.

Our work deals with capacity on the vehicles as Zolfaghari *et al.* (2004) do, where the authors minimize the waiting time of passengers at every stop by taking into account the variance between the departure times. These authors propose heuristics to circumvent the complexity of the proposed model. Puong y Wilson (2008) propose a non-linear mixed-integer linear programming for a real-time disruption response model with emphasis on the train holding strategy. In Delgado *et al.* (2009) and Delgado *et al.* (2012) the aim is to minimize the total waiting times experienced by passengers in the system using a quadratic model.

Our work aims at maintaining congruent headways considering capacity on the vehicles, and in doing so, we expect to reduce passenger-waiting times in the bus corridors. We improve the work of Delgado *et al.* (2012) by reducing the number of variables in the model and the number of times the model is used in real-time scenarios, obtaining exact solutions for the holding times. Moreover, in order to reduce the waiting times of the passengers we bound the holding times of the buses. Another advantage of our proposal is that it adapts easily to the cases where the headways are equal or different during different planning horizons along the day.

3.4 METHODOLOGY AND APPROACH

As mentioned earlier, the core of our methodology consists on interleaving optimization and real-time data retrieving of the bus lines in a rolling horizon planning. The optimization phase of our approach builds and solves efficiently a linear model to maintain congruent headways along the bus line. Our model is used every given time interval[1] to decide how long the buses should be held at the bus stops. Our model requires a real-time data estimation of the state of the system to operate. Such data is provided by the real-time retrieving phase, which in our case of study it is supported via simulation. The simulation of the system provides data related to position of the buses, number of passengers aboard of each bus, and the number of passengers waiting at the stops to build our model.

More precisely, the Bus Bunching Problem, BBP, consists of K buses, each with its own capacity and speed that serve all S stops of a single bus corridor. We can see in Figure 3.3 that each bus k leaves the depot according to an established timetable, serving stops 1 to S before coming back to the depot where all remaining passengers must alight. Notice that overtaking is not permitted. For the optimization phase, we consider that travel times between stops, and λ_s (passengers arrival rate per minute) are deterministic during the period of interest. Moreover, each stop has a dwell time function depending linearly on the number of passengers that board ($boardT$ minutes per passenger).

The characteristics of the line are as following. Parameter cap_k corresponds to the capacity of bus k, $dist_s$ is the distance in meters between stops s and $s-1$, $speed_{ks}$ is the operating speed in meters per minute of bus k between stops s and $s-1$ while the bus is moving, and $OD_{kss'}$ is the fraction of passengers boarding bus k at stop s whose destination is stop s' (for all $s < s'$). The headway between buses k and $k-1$ in this line must be between the interval $[minHead_k, maxHead_k]$ to be considered congruent, which is specified as an input parameter for our model.

[1]The time interval is a parameter in our model, that could be specified by the control unit of the bus company.

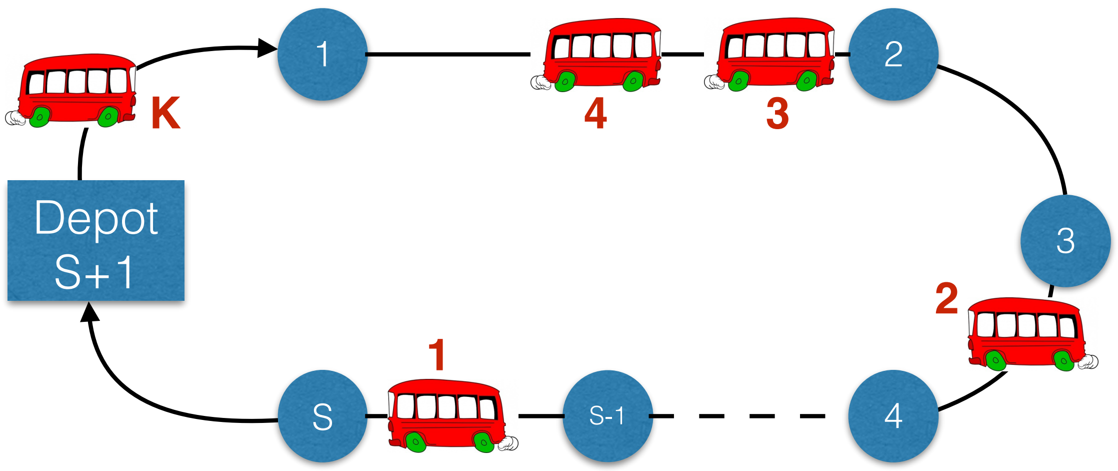

Figure 3.3: Transit bus line model: each bus k leaves the depot according to an established timetable, serving stops 1 to S before coming back to the depot where all remaining passengers must alight.

At time t^0, instant when the holding decisions are needed, we assume that we have the following state of the transit corridor:

- d_k^0 distance between bus k and its last visited stop at time t^0. If the bus is still at a stop, then $d_k^0 = 0$.
- $s(k)$ indicates the last stop that bus k has visited at time t_0. If bus k is at stop s', then $s(k) = s' - 1$. In Figure 3.3, $s(2) = 3$ and $s(3) = 1$, and to simplify the notation, $s(K) = 0$, but $s(1) + 2 = S + 1$.
- c_s^0 is the number of passengers waiting at stop s at time t^0.

Decision variables of our model are the holding times for each bus k at control point s, denoted by h_{ks}. There are auxiliary variables that depend on h_{ks} like the departure times of bus k at stop s that is denoted as td_{ks}. If the departure times at stop s of buses k and $k-1$ is between $[minHead_k, maxHead_k]$, then we consider that they are complying with the established headways. Nevertheless, if this difference of departure times is outside this interval, then we use the concepts of *earliness* and *tardiness* that is frequent in just-in-time scheduling theory (Rios-Solis y Sourd, 2008; Rios-Solis, 2008; Sourd y Kedad-Sidhoum, 2003; Ríos-Mercado y Ríos-Solís, 2012). The *earliness of the headway* between buses k and $k-1$ at stop s is defined as $E_{ks} = \max(minHead_k - (td_{ks} - td_{k-1s}), 0)$ which can be linearized as follows:

$$E_{ks} \geq minHead_k - (td_{ks} - td_{k-1s}), \quad k = 2, \ldots, K, s = s(k)+1, \ldots, S \tag{3.1}$$

$$E_{ks} \geq 0, \quad k = 2, \ldots, K, s = s(k)+1, \ldots, S. \tag{3.2}$$

While the *tardiness of the headway* is $T_{ks} = \max((td_{ks} - td_{k-1s}) - maxHead_k, 0)$ which is equivalent to

$$T_{ks} \geq (td_{ks} - td_{k-1s}) - maxHead_k, \quad k = 2, \ldots, K, s = s(k)+1, \ldots, S \tag{3.3}$$

$$T_{ks} \geq 0, \quad k = 2, \ldots, K, s = s(k)+1, \ldots, S. \tag{3.4}$$

Then, the objective function of BBH is the minimization of the sum of all early and tardy headways:

$$\min \sum_{k=2}^{K} \sum_{s=s(k)+1}^{S} \psi E_{ks} + \varepsilon T_{ks}, \tag{3.5}$$

where ψ and ε are linear penalization for the earliness and the tardiness, respectively, subject to constraints (3.1)-(3.4). Additionally,

The departure times of each bus k at each stop s are defined with two different set of restrictions. The first one is the case where the bus k at time t^0 is between stops $s(k)$ and $s(k)+1$ (in Figure 3.3 this case would apply for bus 2 that is between stops 3 and 4). Here, the departure time of k at $s(k)+1$ is the time that needs the bus to arrive to the stop, plus the dwelling time $dwell_{ks(k)+1}$ (that will be computed later) plus the time the model decides that this bus will hold. This situation is reflected by constraints (3.6). The second case is similar but considers that the bus has not yet reached stop $s-1$ (constraints (3.7)). Restrictions (3.8) impose a limit of *maxHold* to each holding time to guarantee a certain traveling time quality of the passengers.

$$td_{ks(k)+1} = t_0 + \frac{dist_{s(k)} - d_k^0}{speed_{ks(k)}} + dwell_{ks(k)+1} + h_{ks(k)+1}, \quad k \in K \tag{3.6}$$

$$td_{ks} = td_{ks-1} + \frac{dist_{s-1}}{speed_{ks-1}} + dwell_{ks} + h_{ks}, \quad k \in K, s = s(k)+2, \ldots, S-1. \tag{3.7}$$

$$h_{ks} \leq maxHold, \quad k \in K \setminus \{1\}, s = s(k)+1, \ldots, s(k-1). \tag{3.8}$$

From the state variables of the system, we can compute the total number of passengers that will be at stop s when bus k will reach this stop, denoted as $pass_{ks}$ in (3.9) and (3.10), as the number of passengers who are actually in the stop plus the ones that will arrive. The number of passengers that will be in bus k at stop s is equal to the passengers that want board bus k, $pass_{ks}$, minus the proportion of the passengers that left the bus before stop s (restrictions (3.11)). This manner, we can compute the dwell times of bus k at s (restrictions (3.12)). Notice that alighting and friction between the passengers that stay inside the bus could be easily included in the last restriction set.

$$pass_{ks} = c_s^0 + \lambda_s(td_{ks} - t_0), \quad k \in K, s = s(k)+1, \ldots, s(k-1) \tag{3.9}$$

$$pass_{1s} = c_s^0 + \lambda_s(td_{1s} - td_{Ks}), \quad s = s(K)+1, \ldots, S \tag{3.10}$$

$$passBus_{ks} = \min\left(\sum_{i=1}^{s-1} pass_{ki}\left(1 - \sum_{j=i+1}^{s-1} OD_{kij}\right), cap_k\right), \quad k \in K, s = s(k)+1, \ldots, S \tag{3.11}$$

$$dwell_{ks} = passBus_{ks} boardT, \quad k \in K, s = s(k)+1, \ldots, S. \tag{3.12}$$

The following restrictions are the different cases that need to be considered in order to avoid bus overtaking:

$$td_{ks} - td_{k-1s} \geq 0, \quad k \in K \setminus \{1\}, s = s(k-1)+1, \ldots, S \tag{3.13}$$

$$td_{1s} - td_{Ks} \geq 0, \quad s = s(k)+1, \ldots, s(1) \tag{3.14}$$

$$td_{k-1s} - td_{ks} \geq 0, \quad k \in K \setminus \{1\}, s = s(k)+1, \ldots, s(k-1). \tag{3.15}$$

The LP for BBP is then

$$\begin{array}{ll} \min & \sum_{k=2}^{K} \sum_{s=s(k)+1}^{S} \psi E_{ks} + \varepsilon T_{ks} \\ \text{s.t.} & (3.1)-(3.4) \\ & (3.6)-(3.15) \\ & E_{ks}, T_{ks}, h_{ks} \geq 0, \quad k \in K, s \in S. \end{array}$$

Notice that all variables are required to be positive but not integer, so LP can be solved by the simplex method or by a polynomial barrier algorithm. Indeed, main variables h_{ks} represent a time interval so we can consider them as continuous variables. One of the main advantage of LP, besides the fast computational times, is that we could use linear programming sensitivity analysis. Nevertheless, the holding times that are going to be transmitted to the drivers at the bus stations should be seconds or in minutes. Therefore, the model would contain integer variables making it an integer linear programming that can be solved in a exact manner by a branch-and-bound algorithm. Preliminary results showed no drastic increment of the computing times when bus holding variables h_{ks} are integer.

Our model improves and differs the model of Delgado *et al.* (2012) in the following aspects.

- Our objective function is linear so we can obtain optimal solutions for our model.
- The departure times of the buses are according to their established headway or timetable. Only perturbations that arise along the trip are taken into account.
- We only take into account the possible holding times of a bus from its actual position up to the depot instead of consider the holding times for all stops. This reduces the number of variables and makes the problem more realistic.
- We bound the amount of time that a bus can be held at a stop.
- We may have different headways for every pair of buses. This way, recent synchronization timetables can be benefited by our approach and dealing with different planning periods (e.g., rush hour, night time) is natural.
- We do not need to call the model every time a bus arrives at a stop, we can do it each fixed interval of time. This fact is more realistic for a bus company. In our case of study, the company retrieves data of the buses every two minutes.

3.5 EXPERIMENTAL RESULTS

The BBP LP model described in the previous section needs data to be populated. Data can be retrieved through the use of monitoring technologies like Global Positioning Systems (GPS) and Automatic Vehicle Location systems (AVL) in real-time during the execution of the bus corridor. However, to study the impact of our model under different scenarios in the traffic corridor we consider a discrete event simulation.

The single corridor is simulated using the discrete event and stochastic simulator ExtendSim AT version 9.0 (Krahl, 2009; Diamond *et al.*, 2010). The simulator triggers an event every fixed amount of time, in which the positions of the buses and their loads, and the passengers waiting at the stops, together with their traveling destinations, are updated.

Our BBP LP model uses deterministic functions to forecast demands and travel times. Nevertheless, we use stochastic processes in the simulation to reflect a real system. We use a single corridor of 10 kilometers with 30 stops and one depot uniformly distributed like in Delgado *et al.* (2012). There are only 30 stretches, since the last stop is merged with the depot. Travel times of the buses between each pair of stops are distributed as Lognormal with a mean of 0.77 minutes and variance of 0.4 (Hickman, 2001; Zhao *et al.*, 2003). At each stop, passengers arrive randomly using a Poisson distribution with rate equal to one (Jolliffe y Hutchinson, 1975). The mean of the distributions are the parameters used by our model.

When passengers arrive at a bus stop, a destination is assigned to them. Passengers wait in line to board the bus in a first-in/first-out manner. Boarding and alighting times of passengers is set to 2.5 and 1.5 seconds respectively, since all buses have two doors, one for boarding and one for alighting. If passengers cannot enter a bus because it reached its capacity, they will wait in the stop until the next bus with free space arrives. This waiting time is denoted as W_{first}. The headway time windows are set to $[minHead_k, maxHead_k] = [0.3, 0.46]$ minutes for all the buses. Remark that these time windows are easily adjustable for the cases where there are different periods along the day, and for the synchronization timetables that favor transfers. We can measure the waiting and travel times of the passengers and the buses in the simulation since we have modeled these structures as individual agents.

We use a fleet of 60 buses with a maximum capacity of 100 passengers per bus. Every fixed amount of time *interval*, we determine the actions that should be followed by creating the BBP LP model in Java, and solving it with the linear package of Gurobi 5.6. The solution generated contains the holding times for all the buses for all the future stops up to the depot. If after a time *interval* a new solution is generated, then the holding times are updated using a rolling horizon scheme.

Even if we base our scenarios on the ones generated by Delgado *et al.* (2012), there is not a fair comparison since our methodologies consider different assumptions. Nevertheless, we can observe that our approach indeed improves the overall waiting and travel times of the passengers.

The scenarios for the simulation are divided in two parts: *time interval* scenarios and the *parameters setting* scenarios; and they are described in the next subsections.

3.5.1 TIME INTERVAL SCENARIOS

The aim of the time interval scenarios is to determine the optimal policy for controlling when new holding times must be computed and given to the system.

In our case of study for the city of Monterrey, México, the bus company updates every two minutes the positions and all the related data of the buses in the transit corridor. Following this policy, Table 3.1 shows the time interval scenarios in which we test our approach. The first column in Table 3.1 identifies the scenarios while the second column sets the time intervals (in minutes) in which our BBP LP model is constructed and solved to introduce the resulting holding times to the system. We vary these control values from 2 to 10 minutes. Scenario TI_0 does not have any control, and we use it as a baseline to compare the performance of our BBP LP model. The third column is an indicator if restriction (3.8) is applied; that is, if the holding times are bounded. For these scenarios, we set the earliness and the tardiness penalties $\psi = \varepsilon = 1$. The fourth column, W_{first}, corresponds to the total average waiting time (in minutes) of a passenger to board a bus. The fifth column (Travel) represents the total average travel time of passengers in minutes, while the column Pass indicates the average number of passengers in the system during the simulation time. Last two columns indicate the normalized waiting and travel times of each passenger.

Table 3.1: *Time interval* scenarios with earliness and tardiness penalties $\psi = \varepsilon = 1$.

Scen	Control	$maxHold$	W_{first}	Travel	Pass	W_{first}/	Travel/
	(min)	(min)	(min)	(min)		Pass	Pass
TI_0	X	X	1798.0	12035.8	1713.3	1.0	7.0
TI_1	2	X	1115.88	17045.40	1703.1	0.66	10.01
TI_2	5	X	1136.92	18256.20	1746.2	0.65	10.45
TI_3	7	X	1222.66	18907.13	1705.8	0.72	11.08
TI_4	10	X	1362.68	18652.68	1708.5	0.80	10.92
TI_5	2	0.38	1219.52	13112.15	1721.4	0.71	7.62
TI_6	5	0.38	1330.89	13171.48	1737.4	0.77	7.58
TI_7	7	0.38	1463.25	12851.01	1725.6	0.85	7.45
TI_8	10	0.38	1424.52	12450.34	1697.7	0.84	7.33

Ten simulation runs were executed for every scenario, each of them corresponding to one hour of bus operations. Each run has the same initial conditions initialized with random numbers. At the beginning of the simulation the buses are placed evenly spaced along the corridor. For each simulation run, we let the system to evolve freely for five minutes before making any holding. Indeed, five minutes is enough to observe several bus bunching situations to arise.

We observe an increase in the passenger riding time, and potentially operation costs because of the introduction of holding times in the corridor. This behavior is expected, and in concordance with other works (Furth y Muller, 2007). Nevertheless, the passenger-waiting times for the first bus are always reduced, which in fact it is what we wanted to show in first place. Indeed, by controlling the headway we can also control the passenger-waiting times, without the need of using a quadratic objective function in the model.

We can also observe that the best passenger-waiting times are for the cases where the holding controls are applied every 2 to 5 minutes, and without the bounds on the holding times. However, the bounds on the holding times induce a reduction on the travel times, which is an important asset. Figure 3.4 shows the differences in performance when the control (3.8) ($maxHold$) is applied. It shows the percentage of increase on the passenger-waiting times when bounds are applied and the percentage of increase on the travel times when they are not applied. As mentioned, we observe that even if there is an increase on the passenger waiting times when the holding times are bounded, the benefit on the passenger travel times is considerable. Then, maintaining congruent headways reduces the overall travel time of passenger along the whole network.

For a bus company, the less the traffic controller has to give holding orders to the system (i.e., to the bus drivers), the better. Therefore, from Table 3.1 and Figure 3.4, we conclude that the best policy is to consider bounds on the holding times, and apply the controls to the system every 5 minutes, like in the TI_6 scenarios.

In Figure 3.5, we show two histograms of the length of the holding times (x-axis in minutes) for the time interval scenarios with earliness and tardiness penalties $\psi = \varepsilon = 1$, and a control of 5 minutes with and without bounds on the holding times. On the y-axis, we have the frequency the BBP LP model is called for all the simulations of class TI_6. Notice that not all of the holding times are applied, since the rolling horizon may modify several of them. The case when there are limits on the holding times shows that the model either chooses to apply the holding times close to these limits, or not to apply them at all. This is an implicit benefit for the users, and for the traffic controller.

The aim of the BBP model is to reduce bus bunching by maintaining congruent headways. To graphically show that this behavior is being improved by our model, we

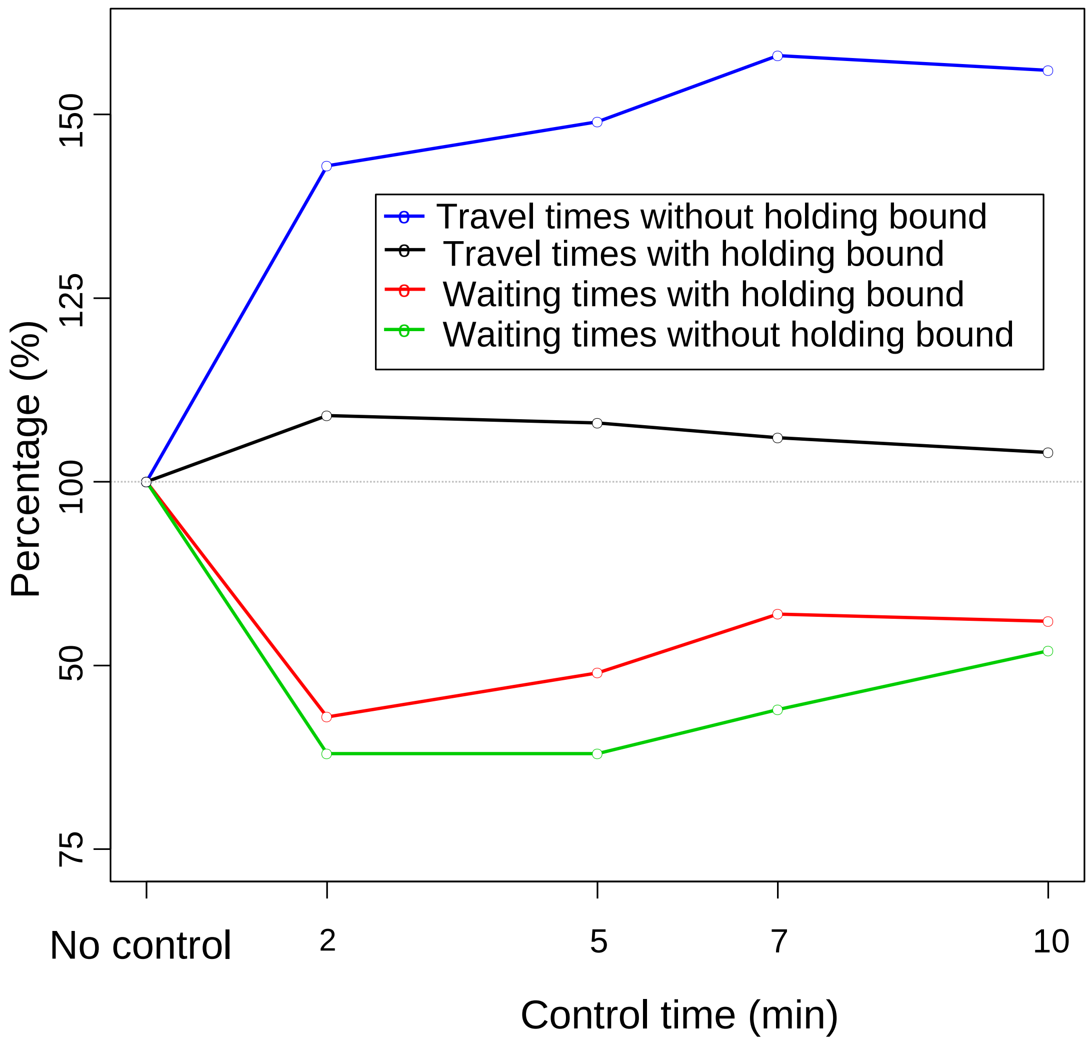

Figure 3.4: Decrease on the waiting times and increase of the travel times for the *time interval* scenarios with earliness and tardiness penalties $\psi = \varepsilon = 1$ and applying bound to holding time.

present Figures 3.6-3.9 for the scenarios with bounds on the holding times. The x-axes in these graphs correspond to time (in minutes), while the y-axes represent stops. Each line in these graphs represents a bus that departs from the depot and cruises all the bus stops. Recall from Section 3.2 (see Figure 3.1) that in the ideal case, we would have *parallel* lines. Figure 3.6 displays the case without control and shows that the simulation makes an stochastic scenario. Here the bus bunching problem is notorious, since there are white

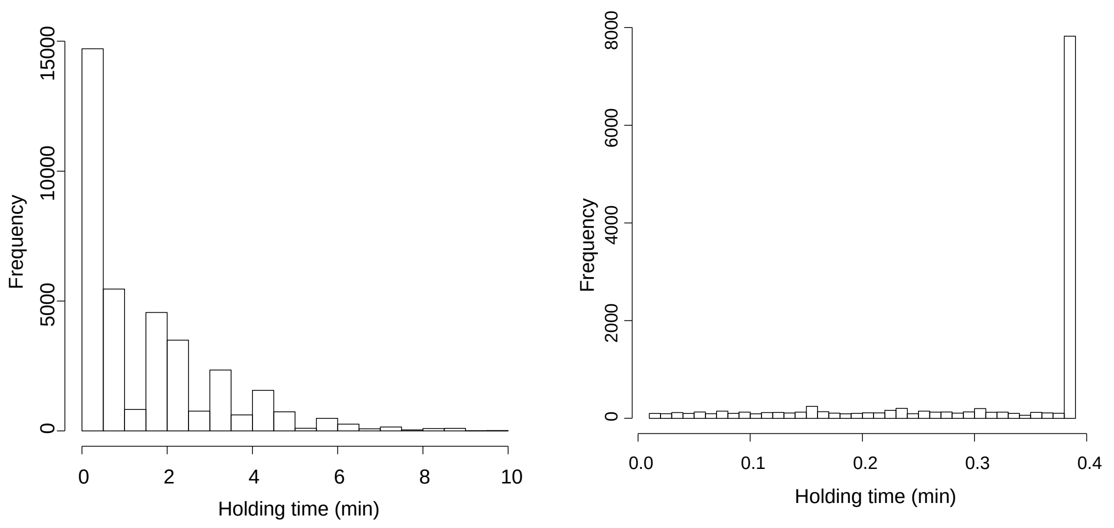

Figure 3.5: Holding times histogram without bounds (left histogram) and with bounds (right histogram) for the *time interval* scenarios with earliness and tardiness penalties $\psi = \varepsilon = 1$, and a control of 5 minutes.

gaps between the lines. Figures 3.7, 3.8, 3.9, have time interval controls of 2, 5, and 7 minutes, respectively. We can observe that with 2 and 5 minutes controls the BBP is reduced, while for control intervals of 7 minutes the BBP appears again.

Figure 3.10 shows two histograms that have in their x-axes the round time of a bus trip. An aspect that we noticed from Table 3.1 is that the travel times increase with the BBP model. This is obvious because the BBP model introduces holding times for the buses in the corridor. Nevertheless, Figure 3.10 shows that the standard deviation when BBP is applied every five minutes (right histogram) is reduced with respect to the case where no controls are used (left hand side histogram).

3.5.2 Parameter setting scenarios

Our next set of experiments modify the earliness ψ and tardiness ε parameters of the BBP LP objective function to observe the impact they have in the passenger-waiting times and

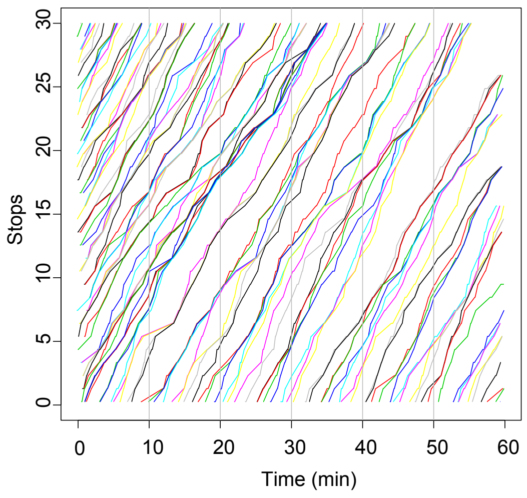

Figure 3.6: Bus transit behavior without control.

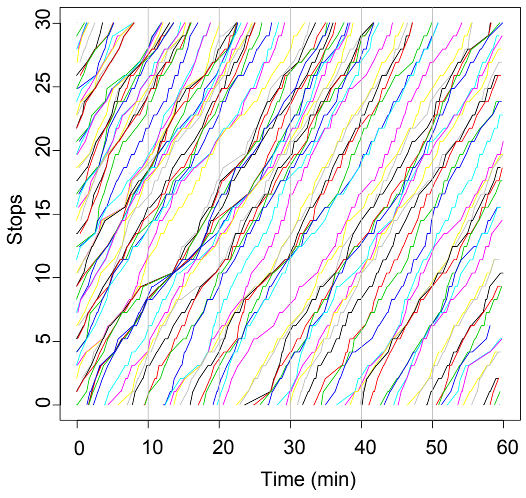

Figure 3.7: Transit with control every 2 min.

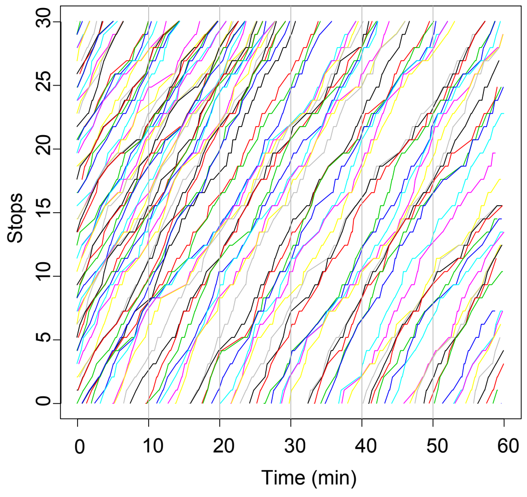

Figure 3.8: Transit with control every 5 min.

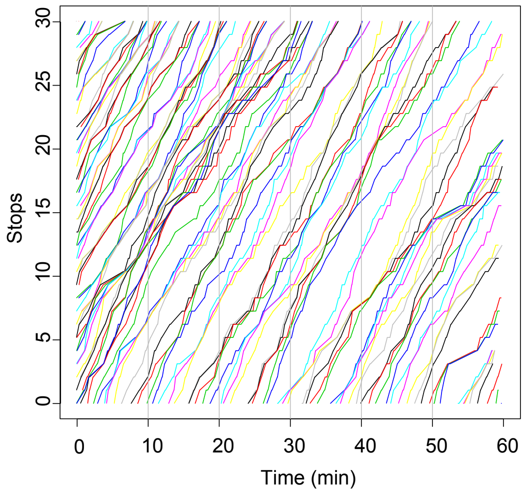

Figure 3.9: Transit with control every 7 min.

travel time. We can see this set of experiments in Table 3.2. The first column in the table identifies the scenarios. Ten simulation runs were considered per scenario. The second column represents the values of the earliness parameter, while the third one corresponds to the tardiness one. The column “Board” denotes the average time (in seconds) a passenger takes to board a bus, while $maxHold$ stands for the time (in minutes) that the holding times are bounded. This table shows the percentage of reduction in passenger waiting

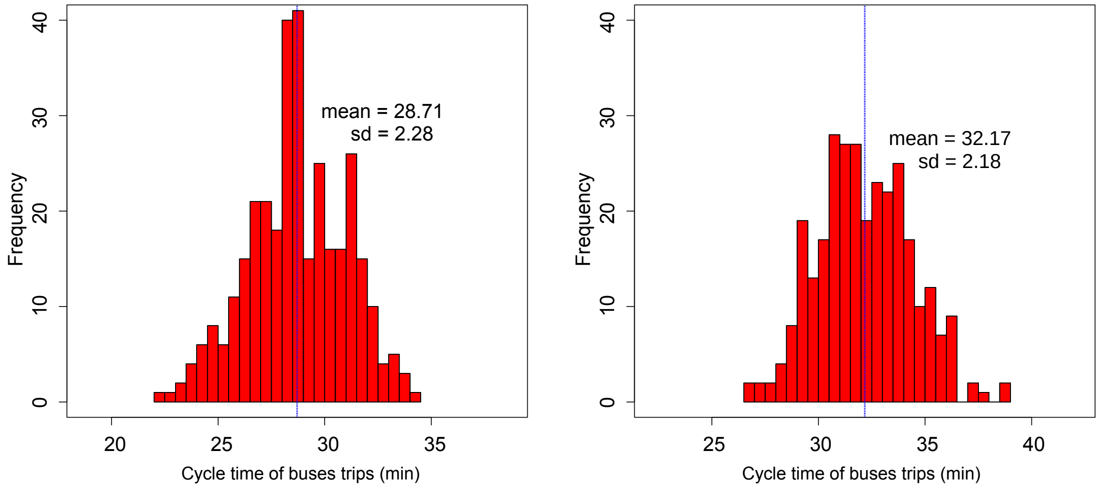

Figure 3.10: Histogram of travel cycle without control (left) and with control interval of 5 minutes (right).

times (W_{first}), and the percentage of increase on the travel times (Travel). Finally, the last column represents the addition of the last two values. Indeed, if there is a reduction on this last column, the percentage would be negative.

An interesting observation from these results is that if we reduce the earliness parameter, we obtain the best results with respect to the passenger-waiting and travel times. Moreover, the BBP LP model yields better results when the holding times are limited by 0.19 minutes, which is also a quality asset for the user.

An statistical analysis confirms the observations from Table 3.2. The most influential parameters are the earliness penalty and the *maxHold* limit. In Table 3.3, we show a linear regression of the parameters studied in this section. The first column is the parameter, the second one corresponds to the "Estimate", the third is the standard error, the fourth stands for to the t value, and the fifth one is the significance.

Table 3.2: Improve of the behavior of waiting time and travel time managing parameters.

			Board	$maxHold$	W_{first}	Travel	W_{first} +Travel
Scen	ψ	ε	(sec)	(min)	% reduction	% increase	% increase
P_1	0	1	1.25	0.19	19%	-2%	-4%
P_2	0	1	1.25	0.38	22%	1%	-2%
P_3	0	1	2.5	0.19	22%	-4%	-6%
P_4	0	1	2.5	0.38	25%	-1%	-4%
P_5	0.5	1	1.25	0.19	34%	10%	4%
P_6	0.5	1	1.25	0.38	55%	39%	27%
P_7	0.5	1	2.5	0.19	37%	11%	5%
P_8	0.5	1	2.5	0.38	56%	41%	28%
P_9	1	0	1.25	0.19	40%	11%	5%
P_{10}	1	0	1.25	0.38	59%	42%	29%
P_{11}	1	0	2.5	0.19	44%	12%	5%
P_{12}	1	0	2.5	0.38	63%	48%	34%
P_{13}	1	0.5	1.25	0.19	36%	10%	4%
P_{14}	1	0.5	1.25	0.38	54%	41%	28%
P_{15}	1	0.5	2.5	0.19	39%	11%	5%
P_{16}	1	0.5	2.5	0.38	57%	40%	27%
P_{17}	1	1	1.25	0.19	39%	11%	5%
P_{18}	1	1	1.25	0.38	48%	27%	17%
P_{19}	1	1	2.5	0.19	39%	56%	43%
P_{20}	1	1	2.5	0.38	57%	50%	36%

	Estimate	Std. Error	t value	Pr($>|t|$)
(Intercept)	1.0314	0.0928	11.11	0.0000
ψ	-0.2234	0.0489	-4.57	0.0004
ε	0.0273	0.0489	0.56	0.5848
Board	-0.0845	0.0646	-1.31	0.2106
$maxHold$	-0.2956	0.0646	-4.57	0.0004

Table 3.3: Linear regression on the main parameters of the BBP model.

3.6 CONCLUDING REMARKS

In this paper, we presented a methodology based on interleaving optimization and real-time retrieving data to maintain congruent headways in a bus corridor with the aim of solving one of the most annoying problems in public transit networks, the Bus Bunching Problem (BBP).

During the optimization phase of our approach, a linear programming model is built and solved to determine the optimal holding times of the buses at the stops to avoid bus bunching. Our model requires real-time data of the state of the system to operate. Such data is provided by the real-time retrieving phase of our approach, which in our case is supported via simulation. The simulation phase of the system provides data related to positions of the buses, number of passengers in the buses, current bus capacities, and number of passengers waiting at the stops to build our model.

One of main the advantages of considering simulation in our methodology is the evaluation of multiple parameters to assess the impact of them in our BBP linear programming model. Therefore, we presented a comprehensive evaluation of such parameters, and found that applying holding controls just every 5 minutes, and bounds on the holding times reduce not only bus bunching frequency but also passenger-waiting times.

We also discussed that most of the works in the literature minimize passenger waiting times, or the variance in the departure times of the buses using quadratic optimization functions, which are more complex to solve. Instead, the linear programming model of our approach makes it suitable for returning optimal solutions efficiently and for interleaving the optimization and real-time retrieving data phases in real-time scenarios.

Although, we observe an increase in the travel time of passengers given the introduction of holding times for the buses in the corridor, our approach performs better (i.e., less passenger-waiting time and acceptable travel time) than no introducing any control into the system. Part of our future work will consider the introduction of other actions into our models to reduce the travel time of the passengers in the corridor and lower operational costs. Particularly, we believe that the introduction of bus overtaking actions (i.e., skipping stops) will balance the total time a passenger spends into the system.

CHAPTER 4

CONCLUSIONS

In this chapter we discuss the general conclusions of the main subjects covered in this dissertation. We also present the main contributions made in the subjects covered along this work, and lastly we present the future work yielded from this research.

4.1 CONCLUSIONS FOR PRODUCT-PIECES-MOLD-MACHINE PROBLEM

As mentioned, the Product-Pieces-Mold-Machine problem is about the manufacturing of a set of pieces that assemble finished products, each piece has a set of molds that can be used to make it, molds can be assigned to a certain number of machines and, setup times arise each time a new mold is installed on a machine. The aim of this problem is to determine the single period lot production size of each finished product, the piece-mold and the mold-machine assignments such to maximize the benefits of selling the finished products. The key point of our method is that we have dropped the constraints that guarantee a feasible scheduling of the molds on the machines. Without these constraints, our mathematical model becomes linear and from its structure we can derive a a Relax&Fix heuristic and a multistart greedy algorithm.

4.2 CONCLUSIONS FOR BUS HOLDING FOR A SIMULATED TRAFFIC NETWORK

As stated in Section 3.6, we presented a methodology based on interleaving optimization and real-time retrieving data to maintain congruent headways in a bus corridor with the aim of solving one of the most annoying problems in public transit networks, the Bus Bunching Problem (BBP).

During the optimization phase of our approach, a linear programming model is built and solved to determine the optimal holding times of the buses at the stops to avoid bus bunching. Our model requires real-time data of the state of the system to operate. Such data is provided by the real-time retrieving phase of our approach, which in our case is supported via simulation. The simulation phase of the system provides data related to positions of the buses, number of passengers in the buses, current bus capacities, and number of passengers waiting at the stops to build our model.

One of main the advantages of considering simulation in our methodology is the evaluation of multiple parameters to assess the impact of them in our BBP linear programming model. Therefore, we presented a comprehensive evaluation of such parameters, and found that applying holding controls just every 5 minutes, and bounds on the holding times reduce not only bus bunching frequency but also passenger-waiting times.

We also discussed that most of the works in the literature minimize passenger waiting times, or the variance in the departure times of the buses using quadratic optimization functions, which are more complex to solve. Instead, the linear programming model of our approach makes it suitable for returning optimal solutions efficiently and for interleaving the optimization and real-time retrieving data phases in real-time scenarios.

Although, we observe an increase in the travel time of passengers given the introduction of holding times for the buses in the corridor, our approach performs better (i.e., less passenger-waiting time and acceptable travel time) than no introducing any control into the system. Part of our future work will consider the introduction of other actions into our models to reduce the travel time of the passengers in the corridor and lower operational costs. Particularly, we believe that the introduction of bus overtaking actions (i.e., skipping stops) will balance the total time a passenger spends into the system.

4.3 FUTURE WORK

In this section we mention the future work derived from the two main subjects covered in this dissertation.

4.3.1 FUTURE WORK DERIVED FROM THE PRODUCT-PIECES-MOLD-MACHINE PROBLEM

There are several lines of investigation pending to consider in regard with the Product-Pieces-Mold-Machine problem. The first one is the consideration of the amount of lifts and cranes due that they are a limited resource, so it will be interesting to include them as part of the problem. The above will increase the problem complexity and probably the separation of the two stages of lot sizing and mold-machine scheduling could not be feasible anymore as the original problem so new separation methods have to be researched.

A second line of investigation is to see what will happen if we consider several periods. The Product-Pieces-Mold-Machine multiperiod problem will have a different structure and a new integer programming model in addition with new solution alternatives like heuristic methods will have to be adapted to this new version of the problem.

The make span is another issue to deal with if the factory already has its production and their main objective is to decrease the time in which these products are going to be produced.

Lastly, it is also important to consider what could happen in a changing environment, for example, when a machine brakes down, a mold is in maintenance, or an urgent order arrives, and how we could adapt our methodology to these changing environments.

4.3.2 FUTURE WORK THE BUS HOLDING FOR A SIMULATED TRAFFIC NETWORK

We are currently working in the research of the bus bunching problem for a problem that has several branches used in one line. This generates a greater amount of randomness in the quantity of passengers as well as their distribution along the line.

We also have to consider that the urban transport systems could have different events which could trigger greater bus-bunching.

Other line of investigation is a lineal programming model that tries to decrease total the travel time of the passengers.

Another important goal of the urban transport system is to rapidly estimate the origin-destination tables in the bus operation models.

List of Figures

LIST OF TABLES

Bibliography

Bartholdi III, J. J. y D. D. Eisenstein (2012), ≪A self-coördinating bus route to resist bus bunching≫, *Transportation Research Part B: Methodological*, **46**(4), págs. 481–491.

Beraldi, P., G. Ghiani, A. Grieco y E. Guerriero (2008), ≪Rolling-horizon and fix-and-relax heuristics for the parallel machine lot-sizing and scheduling problem with sequence-dependent set-up costs≫, *Computers & Operations Research*, **35**(11), págs. 3644–3656.

Ceder, A. (2001), ≪Bus timetables with even passenger loads as opposed to even headways≫, *Transportation Research Record: Journal of the Transportation Research Board*, **1760**(1), págs. 3–9.

Ceder, A. (2007), *Public Transit Planning and Operation: Theory, Modeling and Practice*, Elsevier, Butterworth-Heinemann.

Chacón Mondragón, O. L., O. J. Ibarra Rojas, Y. A. Ríos-Solís y M. A. Saucedo Espinosa (2012), ≪Programación pieza-molde-máquina en planeación de la producción≫, *CIENCIA-UANL*, **15**(58), págs. 59–65.

Chekuri, C. y S. Khanna (2005), ≪A polynomial time approximation scheme for the multiple knapsack problem≫, *SIAM Journal on Computing*, **35**(3), págs. 713–728.

Chen, J.-F. y T.-H. Wu (2006), ≪Total tardiness minimization on unrelated parallel machine scheduling with auxiliary equipment constraints≫, *Omega*, **34**(1), págs. 81 – 89.

CHEN, Z.-L. y W. B. POWELL (2003), «Exact algorithms for scheduling multiple families of jobs on parallel machines», *Naval Research Logistics*, **50**(7), págs. 823–840.

CHRÉTIENNE, P., Ö. HAZIR y S. KEDAD-SIDHOUM (2011), «Integrated batch sizing and scheduling on a single machine», *Journal of Scheduling*, **14**(6), págs. 541–555.

CORMEN, T. H. (2009), *Introduction to algorithms*, MIT press.

DAGANZO, C. F. (2009), «A headway-based approach to eliminate bus bunching: Systematic analysis and comparisons», *Transportation Research Part B: Methodological*, **43**(10), págs. 913–921.

DAGANZO, C. F. y J. PILACHOWSKI (2011), «Reducing bunching with bus-to-bus cooperation», *Transportation Research Part B: Methodological*, **45**(1), págs. 267–277.

DELGADO, F., J. C. MUNOZ y R. GIESEN (2012), «How much can holding and/or limiting boarding improve transit performance?», *Transportation Research Part B: Methodological*, **46**(9), págs. 1202–1217.

DELGADO, F., J. C. MUÑOZ, R. GIESEN y A. CIPRIANO (2009), «Real-time control of buses in a transit corridor based on vehicle holding and boarding limits», *Transportation Research Record: Journal of the Transportation Research Board*, **2090**(1), págs. 59–67.

DESAULNIERS, G. y M. HICKMAN (2007), «Public transit», *Transportation, Handbooks in Operations Research and Management Science*, págs. 69–127.

DIAMOND, B., D. KRAHL, A. NASTASI y P. TAG (2010), «ExtendSim advanced techology: integrated simulation database», en *Proceedings of the Winter Simulation Conference*, Winter Simulation Conference, págs. 32–39.

DILLENBERGER, C., L. F. ESCUDERO, A. WOLLENSAK y W. ZHANG (1994), «On practical resource allocation for production planning and scheduling with period overlapping setups», *European Journal of Operational Research*, **75**(2), págs. 275–286.

DING, Y. y S. I. CHIEN (2001), ≪Improving transit service quality and headway regularity with real-time control≫, *Transportation Research Record: Journal of the Transportation Research Board*, **1760**(1), págs. 161–170.

EBERLEIN, X. J., N. H. WILSON y D. BERNSTEIN (2001), ≪The Holding Problem with Real–Time Information Available≫, *Transportation science*, **35**(1), págs. 1–18.

EREN, T. (2007), ≪A Multicriteria Scheduling with Sequence-Dependent Setup Times≫, *Applied Mathematical Sciences*, **1**(58), págs. 2883–2894.

ESCUDERO, L. F. y J. SALMERON (2005), ≪On a fix-and-relax framework for a class of project scheduling problems≫, *Annals of Operations Research*, **140**(1), págs. 163–188.

FURTH, P. G. y T. H. MULLER (2007), ≪Service reliability and optimal running time schedules≫, *Transportation Research Record: Journal of the Transportation Research Board*, **2034**(1), págs. 55–61.

GICQUEL, C., M. MINOUX, Y. DALLERY *et al.* (2008), ≪Capacitated lot sizing models: a literature review≫, Open Access Article hal-00255830, Hyper Articles en Ligne.

HAASE, K. y A. KIMMS (2000), ≪Lot sizing and scheduling with sequence-dependent setup costs and times and efficient rescheduling opportunities≫, *International Journal of Production Economics*, **66**(2), págs. 159–169.

HAZIR, Ö. y S. KEDAD-SIDHOUM (2012), ≪Batch sizing and just-in-time scheduling with common due date≫, *Annals of Operations Research*, págs. 1–16.

HICKMAN, M. D. (2001), ≪An analytic stochastic model for the transit vehicle holding problem≫, *Transportation Science*, **35**(3), págs. 215–237.

IBARRA-ROJAS, O., R. RÍOS-MERCADO, Y. RIOS-SOLIS y M. SAUCEDO-ESPINOSA (2011), ≪A decomposition approach for the piece–mold–machine manufacturing problem≫, *International Journal of Production Economics*, **134**(1), págs. 255–261.

IBARRA-ROJAS, O. J., F. LÓPEZ-IRARRAGORRI y Y. A. RIOS-SOLIS (2014), ≪Multiperiod Synchronization Bus Timetabling≫, *Under review*, **–**(-), págs. –.

IBARRA-ROJAS, O. J. y Y. A. RIOS-SOLIS (2012), ≪Synchronization of bus timetabling≫, *Transportation Research Part B: Methodological*, **46**(5), págs. 599–614.

IBARRA-ROJAS, O. J., Y. A. RIOS-SOLIS y O. L. CHACON (2010), ≪Piece-Mold-Machine Manufacturing Planning≫, en B. Nag (editor), *Intelligent Systems in Operations: Models, Methods, and Applications in the Supply Chain*, IGI Global, Hershey, págs. 105–117.

JOLLIFFE, J. y T. HUTCHINSON (1975), ≪A behavioural explanation of the association between bus and passenger arrivals at a bus stop≫, *Transportation Science*, **9**(3), págs. 248–282.

KRAHL, D. (2009), ≪ExtendSim advanced techology: discrete rate simulation≫, en *Winter Simulation Conference*, Winter Simulation Conference, págs. 333–338.

LI, S., G. LI y S. ZHANG (2005), ≪Minimizing makespan with release times on identical parallel batching machines≫, *Discrete Applied Mathematics*, **148**(1), págs. 127–134.

LIN, C. K. Y., C. L. WONG y Y. C. YEUNG (2002), ≪Heuristic Approaches for a Scheduling Problem in the Plastic Molding Department of an Audio Company≫, *Journal of Heuristics*, **8**(5), págs. 515–540.

MARINELLI, F., M. E. NENNI y A. SFORZA (2007), ≪Capacitated lot sizing and scheduling with parallel machines and shared buffers: A case study in a packaging company≫, *Annals of Operations Research*, **150**(1), págs. 177–192.

MARTELLO, S. y P. TOTH (1990), *Knapsack problems: algorithms and computer implementations*, John Wiley & Sons, Inc.

PLATEAU, M.-C. y Y. A. RIOS-SOLIS (2010), ≪Optimal solutions for unrelated parallel machines scheduling problems using convex quadratic reformulations≫, *European Journal of Operational Research*, **201**(3), págs. 729–736.

PUONG, A. y N. H. WILSON (2008), ≪A train holding model for urban rail transit systems≫, en *Computer-aided Systems in Public Transport*, Springer, págs. 319–337.

QUADT, D. y H. KUHN (2008), ≪Capacitated lot-sizing with extensions: a review≫, *4OR*, **6**(1), págs. 61–83.

RESENDE, M. G. y C. C. RIBEIRO (2014), ≪GRASP: greedy randomized adaptive search procedures≫, en *Search methodologies*, Springer, págs. 287–312.

RÍOS-MERCADO, R. Z. y Y. A. RÍOS-SOLÍS (2012), *Just-in-time Systems*, tomo 60, Springer.

RIOS-SOLIS, Y. A. (2008), ≪Scheduling with earliness–tardiness penalties and parallel machines≫, *4OR*, **6**(2), págs. 191–194.

RIOS-SOLIS, Y. A. y F. SOURD (2008), ≪Exponential neighborhood search for a parallel machine scheduling problem≫, *Computers & Operations Research*, **35**(5), págs. 1697–1712.

SOURD, F. (2006), ≪Dynasearch for the earliness-tardiness scheduling problem with release dates and setup constraints≫, *Operations Research Letters*, **34**(5), págs. 591–598.

SOURD, F. y S. KEDAD-SIDHOUM (2003), ≪The one-machine problem with earliness and tardiness penalties≫, *Journal of Scheduling*, **6**(6), págs. 533–549.

STECCO, G., J.-F. CORDEAU y E. MORETTI (2008), ≪A branch-and-cut algorithm for a production scheduling problem with sequence-dependent and time-dependent setup times≫, *Computers & Operations Research*, **35**(8), págs. 2635–2655.

SUN, A. y M. HICKMAN (2008), ≪The holding problem at multiple holding stations≫, en *Computer-aided systems in public transport*, Springer, págs. 339–359.

VILÍM, P. (2006), ≪Batch Processing with Sequence Dependent Setup Times≫, en P. Van Hentenryck (editor), *Principles and Practice of Constraint Programming - CP 2002*, *Lecture Notes in Computer Science*, tomo 2470, Springer, Berlin, págs. 153–167.

WENG, M. X., J. LU y H. REN (2001), ≪Unrelated parallel machine scheduling with setup consideration and a total weighted completion time objective≫, *International Journal of Production Economics*, **70**(3), págs. 215–226.

ZHAO, J., S. BUKKAPATNAM y M. M. DESSOUKY (2003), «Distributed architecture for real-time coordination of bus holding in transit networks», *Intelligent Transportation Systems, IEEE Transactions on*, **4**(1), págs. 43–51.

ZOLFAGHARI, S., N. AZIZI y M. Y. JABER (2004), «A model for holding strategy in public transit systems with real-time information», *International Journal of Transport Management*, **2**(2), págs. 99–110.

Resumen autobiográfico

M.C. Miguel Lorenzo Morales Marroquín

Candidato para obtener el grado de

Doctor en Ingeniería

con especialidad en ingeniería de sistemas

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Tesis:

LOT PRODUCTION SIZE PROBLEM AND SIMULATION OF URBAN TRANSPORT

I was born on Thursday, March 11, 1982 at 17:30 pm in the city of Monterrey, Nuevo León, México. I am the eldest son of Miguel Morales Garza and Blanca Mireya Marroquín de Morales.

In June 2009 I graduated as a Chemical Engineer from the Facultad de Ciencias Químicas de la Universidad Autónoma de Nuevo León.

On Wednesday 23 September 2009 I was blessed with the birth of my first daughter Regina Morales Suarez at the age of 27 years.

On Monday April 25, 2011 I was blessed for the second time with the birth of my second daughter Minerva Morales Suarez.

In July 2012 I graduated as a Master of Science degree in Systems Engineering from Facultad de Ingeniería Mecanica y Electrica de la Universidad Autónoma de Nuevo León under the supervision of Yasmín Á. Riós Solís and with the support of CONACyT.

In August 2012 I started as a Ph.D. student of the graduate engineering systems of the Facultad de Ingeniería Mecanica y Electrica de la Universidad Autónoma de Nuevo León, also under the supervision of Yasmín Á. Riós Solís and the support of CONACyT.

I currently live in the company of my beloved wife Genesis Suarez Lazalde and my two daughters in Guadalupe, Nuevo León, Mexico.