2009

Scheduling inbound calls in call centers

Somchan Vuthipadadon
Iowa State University

Scheduling inbound calls in call centers

by

Somchan Vuthipadadon

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Industrial and Manufacturing Systems Engineering

Program of Study Committee:
Sigurdur Olafsson, Major Professor
Jo Min
Sarah M. Ryan
Yoshinori Suzuki
Corinne Langinier

Iowa State University

Ames, Iowa

2009

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Scheduling inbound calls, namely assigning calls to Customer Service Representatives (CSRs) and sequencing the calls waiting for each CSR, is a key task in call center operations. In most call center this is achieved using simple priority rules, but in this dissertation we show that performance can be significantly improved by employing an optimization approach. Specifically, we formulate three different Integer Programming (IP) problems for such call scheduling, with objective functions of 1) minimizing the Total Flow Time (TFT), 2) minimizing the Maximum Flow Time (MFT) of any call, and 3) minimizing the Maximum Deviation of Cumulative Assigned Workload (MDCAW) for CSRs. We also report the results of a numerical experiment designed to evaluate under what conditions these IP formulations give superior performance and which objective should be chosen. Our findings indicate that optimization is most valuable under realistic scenarios involving specialized but broadly trained CSRs and high call centre utilization rates. Furthermore, both the flow time and workload related objective functions are found to be useful, depending on the characteristics of the call center and the performance measures that are most important to call center management. We explore several solution techniques such as IP reformulation, Lagrangian relaxation and duality, cutting plane algorithm, and heuristic approach for solving the formulated IPs. For those solution algorithms, the qualities of the solution and the computational times of solving the IPs using a standard solver are compared to signify the effective approaches that make the optimization a competitive approach for scheduling inbound calls. Numerical results show that the heuristics optimization approach is preferable to any other solution investigated in terms of solution quality while the cutting plane algorithm is preferable in terms of computational times.

Additionally, a case study comparing performances of a call center as resulted from using its current routing method with the performance resulted from the suggested solution techniques is presented.

CHAPTER 1. INTRODUCTION

Call centers have become an integral part of doing business as many companies use call centers for customer interactions such as selling products and services, providing product support, and resolving billing issues (Cezik et al., 2001). The three main components of any call center are the customers (or calls), the technologies employed, and the customer service representatives (CSRs). Customers are typically categorized according to the flow direction of the call as either inbound call (or incoming call) or outbound call (or outgoing call). Call centers are categorized accordingly based on the calls they process, that is, as inbound call centers, outbound call centers, or mixed call centers (Gans et al., 2003). Various technologies are deployed to assist the communication process between customers and CSRs. This includes computer telephony integration (CTI), networking hardware, automated call distribution (ACD), private branch exchange (PBX) phone switch, interactive voice response (IVR), and the necessary software (Sharp, 2003).

Since different customers have diverse demands, many call centers have CSRs that are skilled to deal with various types of calls. However, most CSRs are specialized in certain calls while only trained on others, which makes the process of matching calls to the proper CSRs vital to providing efficient service. There is therefore a need for effective methods for scheduling inbound calls, which in most call centers is handled by an ACD system that is used to distribute calls among idle qualified CSRs based on a set of rules (Koole and Mandelbaum, 2002). Selecting the most appropriate method for this task is not trivial. There are numerous simple approaches, such as first come first served (FCFS) and skill based routing (SBR). However it is not possible to find a simple method that will work best for all call centers at all times. Each call center much therefore identify a suitable method based on its characteristics in order to achieve the greatest benefits from both a business and employee viewpoint.

1.1 Problem statement

This research addresses the problem of scheduling inbound calls, namely assigning inbound calls to customer service representatives (CSRs) and sequencing the calls waiting for each CSR in call centers. Our intent is to show that the performance of call centers can be improved by employing an optimization approach for this scheduling problem, and at the same time designing efficient solution algorithms for the problem.

1.2 Research motivation and significance

Nowadays, as we live in a service era, customer satisfaction is viewed as the top key business priorities of most industries. Thus providing "Service when customers need it." has become an important strategy to provide the superior experience and enhance the customer satisfaction. Many companies use call centers to provide effective interactions with their customers. Therefore there is a huge market for call centers all over the world, which continues to grow each year. For most call centers, inbound call scheduling is one of the critical business functions. With an effective call scheduling policy, the call center can not only offer better service, but also the increasing its performance. Hence the development of an approach to provide solutions for call scheduling problem is significance and in fact essential.

This research will provide call centers with alternative approaches for inbound calls scheduling. This will not only be a more systematic approach than previously available, but the proposed approach also provides good solutions which consequently results in improved call center performance. This will benefit both call centers and customers

1.3 Research objective

The main objective of this research is an effective approach for scheduling inbound calls in call centers. To achieve the objective two main tasks are needed: 1) develop an IP formulation to model inbound calls scheduling in call center and, 2) explore solution algorithms to effectively solve those IPs. Precisely what constitutes an effective approach in this context will be defined in Chapter 4.

1.4 Scope of the research

This research deals with the problem of scheduling inbound calls in call centers by first formulating it as an integer programming (IP) problem. Three different objective functions are used: minimizing the total flow time (TFT), minimizing the maximum flow time (MFT), and minimize the maximum deviation of cumulative assigned workload (MDCAW). For each approach call center performances, including the number of calls, the total flow time, the flow time and the waiting time of each call, the amount of workload assigned to each CSR, and the service level, are examined through a numerical experiment. Through the experiment, the preliminary results show that, under various environment setting, the performance of call centers can be significantly improved by employing an optimization approach to schedule inbound calls. Particularly, the optimization approach is most valuable under realistic scenarios involving specialized but broadly trained CSRs and high call center utilization rates. Hence, this motivates the need for the optimization approach.

Secondly, given the IP formulation, this research will explore several solution techniques such as IP reformulation, Lagrangian relaxation and duality, generating valid inequalities (cutting planes), and heuristics for solving the formulated IPs. For each of those proposed solution algorithms, the quality of the solution and the computational time needed for solving the problem are compared.

1.5 Contribution of the research

The contribution of this work is the novel IP formulation of inbound call scheduling, which to the best of our knowledge has not been considered before, and the insights that are obtained by solving the corresponding IPs. In particular, we show that for many call centers it is possible to significantly improve performance by optimizing the inbound call schedule. Furthermore, we characterize call centers where such an optimization approach is the most beneficial and provide managerial insights into what type of optimization problem should be solved based on the call center characteristics and which performance measures are of most interest. Besides the developed formulation, our investigation on the problem-solving performances of the proposed solution algorithms also serves as a guide line for selecting the suitable solution algorithm. It offers an idea of which method will be efficient for solving call scheduling problem based on their values on the performances.

1.6 Summary of subsequent chapters

This dissertation is organized as follows. A research background, consisting of a review of existing call center related literature and an overview of IP, is given in the Chapter 2. Chapter 3 introduces our IP models as instruments to handle inbound calls scheduling task and examines the effectiveness of the models. Chapter 4 discusses and analyzes four purposed solution techniques. Chapter 5 presents a case study examined the potential of using IP approach for call scheduling problem in a real call center. Finally, Chapter 6 describes conclusions and further research directions.

CHAPTER 2. RESEARCH BACKGROUND

This chapter is organized into two sections. The first section provides a review of literature related to call centers. The second section offers an overview of the theoretical knowledge related to Integer Programming (IP), which is the vital approach employed in this paper.

2.1 Review of Call Center Related Literatures

The proliferation of modern technology has changed the way most enterprises interact with customers. Nowadays call centers are used as a bridge to reach customers. Many companies have found that establishing call centers improves of company-customers relationship. This makes call centers an important part of any business.

In the last decade, call centers has been received with considerable enthusiasm by researchers. Numerous research studies that are either directly or indirectly related to call centers have been conducted for more than 50 years, and for a comprehensive and updated bibliography of studies Mandelbaum (2004) is an excellent source. For other introductions to the field, Gans et al. (2003) provide an overview of the telephone call centers and Gans and Zhou (2003) provide a comprehensive tutorial on call-centre operations.

The operations research community has contributed significantly to the call center research literature, and IP approaches for modeling certain aspects of call centers have been studied by various researchers. For example, Henderson and Berry (1976) apply heuristic methods and linear programming for telephone operator shift scheduling. Adel and Pearce (1979) and Bruce and Parson (1993) utilize queuing and IP techniques to determine optimal staffing levels for achieving a designated objective. Berman and Larson (1994) introduce the utilization of the JIT concept with a cost-driven objective to determine the optimal work force schedule. Thompson (1995) presents an IP for developing optimal shift schedules. Mason et al. (1998) develop an integrated approach using simulation, heuristic descent and IP techniques to determine near-optimal staffing levels. Gulati and Malcolm (2001) present a general IP of call scheduling model and use simulation to compare three different solving approaches (i.e., heuristic, batch optimized, and dynamic optimized). Cezik et al. (2001) design a weekly tour scheduling using an application of IP model. Atlason et al. (2002) combine simulation with IP by introducing an iterative cutting-plane algorithm on an integer program, for minimizing the staffing costs of a multi-skill call center subject to satisfying service-level requirements that are estimated using simulation.

As indicated above, IP formulations have been used extensively for work force management, but this is not the case for day-to-day operations in call centers, such as scheduling of inbound calls. However, numerous authors have considered routing of calls, primarily using queuing theory or similar analysis. For example, Melsa et al. (1990) present a neural network solution for call routing through a three stage interconnection network. Levy et al. (1994) propose a call-distributing algorithm that makes effectively use of available information to improve load balancing and SL in the centre. Borst (1995) studies a probabilistic call distribution to minimize a weighted sum of the mean waiting times. Durinovic and Levy (1997) present a routing solution for toll-free customer that was used at an AT&T call distribution centre. The algorithm used provides call-by-call routing to multiple customer sites based on periodic site-state updates and can address various objective goals. Kogan et al. (1997) study call center performance under two alternative call routing strategies, namely FCFS and Minimum-Expected-Delay (MED) and conclude that the performance of using FCFS is perceived as optimal. Marbach et al. (1998) formulate a dynamic programming problem and apply a reinforcement learning method and decomposition approach to find call admission control and routing policies. Gans and Zhou (2003) formulate a call-routing problem as a queuing system with SL constraint to determine the structure of effective routing policies. Finally, Koole et al. (2003) utilize heuristic and queuing approaches to analyze the performance of multi-skill call centers and also to determine optimal skill set for call center employees.

Although we have seen that IP approaches have been widely used in many call center research studies, to the best of our knowledge, such an approach has not been used previously to schedule inbound calls. As previously noted, most research related to routing of calls uses a queuing perspective (e.g., Garnett and Mandelbaum, 2000; Gans and Zhou, 2003; Koole et al., 2003). A review of the use of queuing theory for analyzing call centers can be found in Koole and Mandelbaum (2002). On the other hand, queuing analysis has certain limitation since it may often be difficult to perform (Pinedo et al., 2000), hence the IP approach developed in this paper may serve as an alternative is such situations.

2.2 Overview of IP

The purpose of this section is to briefly summarize some basic IP concept and to describe some of the IP solution techniques, which will be applied in this research for inbound calls scheduling. Good introductions to this subject can be found in numerous textbooks such as Nemhauser and Wolsey (1988), Gottfried and Weisman (1973), and Bertsimas and Tsitsiklis (1997).

2.2.1 Integer Programming Problems

Integer programming problems (IPs) are optimization problems in which some or all variables are required to be integers. The general form of IP model consists of (1) an objective function, either be maximizing or minimizing function (2) inequality and equality constraints, and (3) integrality constraints on some or all of the variables. Regarding the integrality restrictions, IPs can be categorized into two main types: (a) all-integer programming problems, in which all variables are integers, and (b) mixed-integer programming problems, in which only some of the variables are required to be integers.

In this research study, we will concentrate on the all-integer programming problem with the special case where all variables are restricted to be either zero or one, namely 0-1 IP.

$$\min\left\{f(x) : g(x) \leq 0, x \in B^n\right\} \qquad \text{(0-1 IP)}$$

Here f is the objective function, g are the constraints, and B is the set of zero-one (binary) variables.

If the objective function and all constraints are linear, the problem is considered integer linear programming (ILP). Otherwise it is the integer nonlinear programming (INLP), which are considered to be one of the hardest to solve.

The robustness of the 0-1 IP model makes it applicable to many applications, such as assignment and matching problems, scheduling problems, set-covering problems, set-packing problems, set-partitioning problems, fixed-charge network flow problems, capacitated facility location problems, and traveling salesman problems.

2.2.2 Model Formulation and Reformulation

As stated by Nemhauser and Wolsey (1988):

"In integer programming, formulating a "good" model is of crucial importance to solving the model"

While there are many possible ways to mathematically formulate the same problem, those formulations are not at the same level of difficulty to solve. The performance of solving a problem often depends on how the IP is formulated. In IP, bounds on the value of the objective function are often used to determined how "good" the IP formulations are since the efficiency of the formulation is very dependent on the sharpness of the bound (Nemhauser and Wolsey, 1988). Therefore, bounds are often used as a measurement to compare the quality of the formulation. Note that the IP formulation with the tighter bound is the better formulation. A bound (Z_{LP}) can be obtained by solving the linear relaxation of the IP problem. For maximization problem, we compare upper bounds. The better bound or tighter bound refers to the smaller bond value. While for minimization problem, we compare lower bounds and the better bound is the higher bound value. More information on linear relaxation technique is given in the next section.

Example 2-1

Consider, for example, the following IP problem:

$Max\ x_1 + x_2$

s.t. $6x_1 + 4x_2 \leq 15$ (Constraint 1)

x_1, x_2 are integer variables

Let us explore the set of feasible points (S) of this example.

S = {(0,0), (0,1), (0,2), (0,3), (1,0), (1,1) (1,2), (2,0)}

One might easily see that

$3x_1 + 2x_2 \le 7$ (Constraint 2)

also gives the same set of feasible points.
However, which constraint yields the better formulation?

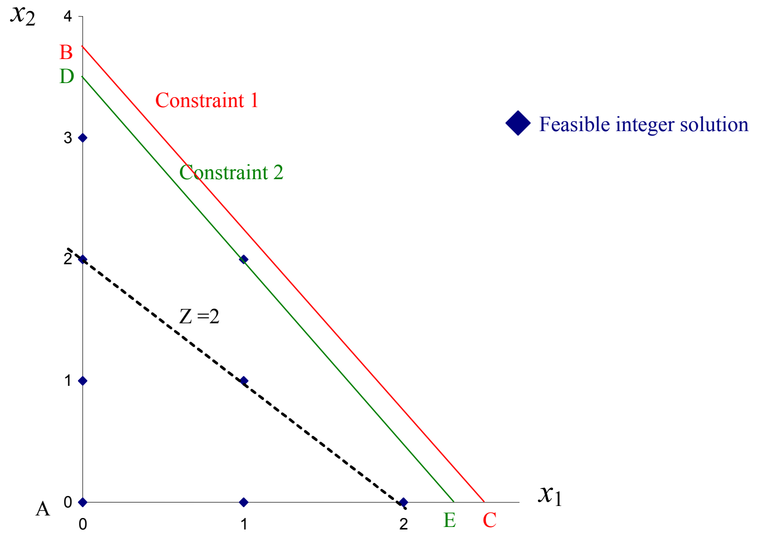

Figure 2-1. Graphical representation of example 2-1

The graphical representation of example 2-1, show in figure 2-1, reveals the answer. Area ABC and ADE are the feasible regions of the problem with constraint 1, and constraint 2, respectively. By replacing constraint 1 with constraint 2, we observe the smaller feasible region. Also, the upper bound with constraint 2 found at point D ($Z^2{}_{LP}$ =3.5) has the better value than that of from constraint 1 found at point B ($Z^1{}_{LP}$ =3.75). Hence, we say that constraint 2 makes the better formulation for this example.

To obtain a good formulation, besides the choice of constraints as illustrated in example 2-1, reformulation of IPs is another bound tightening technique which has gained a lot of attentions from researchers. Various automatic reformulation techniques have been studied and successfully implemented to improve formulation (Guignard and Spielberg, 1981; Williams, 1985; Andersen and Andersen, 1995; Brearley, Mitra and William, 1975; and Hoffman and Padberg, 1991).

Two common reformulation techniques used to improve model efficiency are:

- Adding or removing variables, constraints, while keeping the same model structure.

Example 2-2

Let revisit example 2-1, now not only replace constraint 1 with constraint 2, but let us also add a new constraint.

$x_2 \leq 3$ (Constraint 3)

After adding the new constraint, we find that the new feasible region is smaller (area AFGE) and the upper bound of the new formulation found at point G becomes better ($Z^3{}_{LP}$ $=3.33< Z^2{}_{LP}< Z^1{}_{LP}$). See figure 2-2 for the graphical representation of example 2-2.

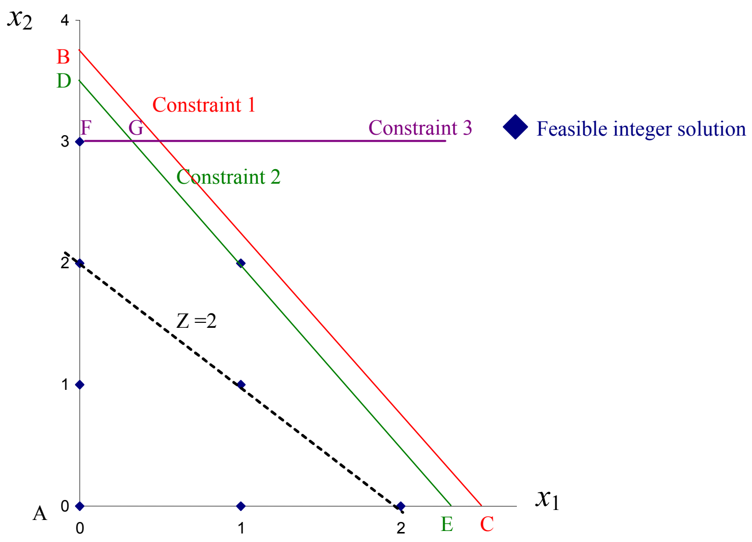

Figure 2-2. Graphical representation of example 2-2

The approach used in example 2-2 is also known as cutting planes method, where one or more additional constraints (or cuts) are introduced to the original problem to cut off a superfluous portion of the previously feasible region (see shade area in figure 2-2). The additional constraint is called a cutting plane. For more information on this method, see Geoffrion and Marsten (1972), Gomory (1958).

Example 2-3

Let us add another constraint to the formulation

$$x_1 \le 2 \qquad \text{(Constraint 4)}$$

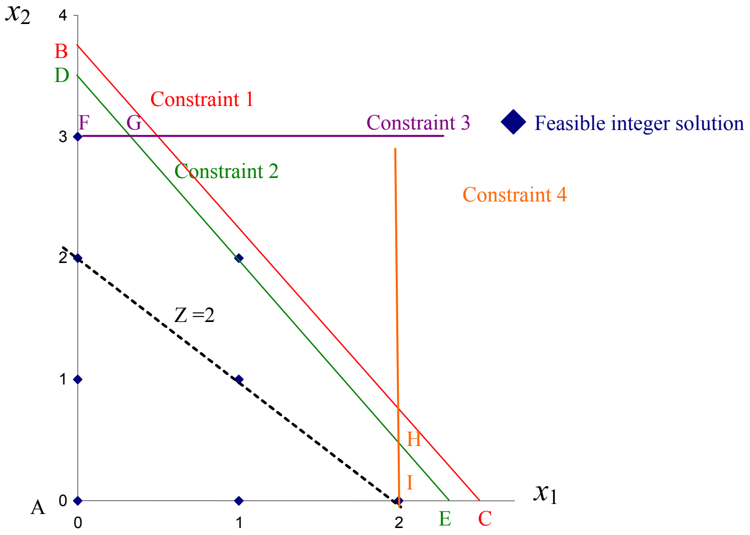

Figure 2-3. Graphical representation of example 2-3

Resolving the LP relaxation, we found Z^4_{LP} =3.33 at point G, which is equal to the bound value in the previous example (Z^4_{LP} = Z^3_{LP} =3.33< Z^2_{LP} < Z^1_{LP}). Although figure 2-3 shows that the new constraint can trim down the feasible region (new feasible region is area AFGHI), the LP solution indicates the unnecessary of adding this new constraint since it does not improve the LP bound.

From this example, we learn that only valid inequalities that can tighter the LP bounds are significant and should be added as new constraints through the reformulation process. Some well-accepted techniques to obtain general valid inequalities are integer rounding, disjunctive constraint, Boolean implication, geometric or combinatorial implications, C-G rounding, fractional cutting-plane algorithm, and Gomory's fractional cuts and rounding (Nemhauser and Wolsey, 1988). However, the valid inequalities obtained by these general techniques are sometimes not very powerful. The more potential techniques have been studies for special problem structure. See Farias and Nemhauser (2001) and Gottlieb and Rao (1990, 1986) for generalized assignment problem, Balas (1975a), Hammer, Johnson and Peled (1975), and Wolsey (1975) for knapsack problem, Grotschel and Padberg (1979) for traveling salesman problem, Leung and Magnanti (1986) and Pochet (1988) for capacitated facility location, Cho et al. (1983a, 1983b) for uncapacitated facility location. More specific detail on each individual technique can be found in Nemhauser and Wolsey (1988).

- Using a different model presentation. This method completely transforms the model by reformulate the problem with the new objective function, variables, and constraints.

An example of the good formulation obtained by defining new set of variables can be illustrated using the following uncapacitated lot sizing (ULS) problem.

Example 2-4

Decide on a production plan for a single product over 4 weeks that will minimize cost while all demands are satisfied. Fixed cost is $300 per week. Unit variable cost of production is $50. Unit holding cost is $3. Demands are 200 units for week 1, 300 units for week 2, 100 units for week 3, and 400 units for week 4.

The problem can be formulated by letting d_i be the demand in week i, by defining x_i, s_i as the amount produced in week i and stock at the end of week i and by defining a binary variable y_i, indicating whether x_i>0 or not. This lead to the following IP:

$$Min \sum_{i=1}^{4}(300y_i + 50x_i + 3s_i)$$

s.t.

$s_{i-1} + x_i = d_i + x_i$ for $i = 1, 2, 3, 4$

$x_i \leq 1000y_i$

$s_0 = 0,\ d_1 = 200,\ d_2 = 300,\ d_3 = 100,\ d_4 = 400$

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

$s_1, x_i \geq 0$

Another possibility formulation is obtained by defining w_{it} as the amount produced in week i to satisfy the demand in week $j \geq i$, and by defining a binary variable y_i as above formulation. This lead to the alternative IP:

$$Min \sum_{i=1}^{4} 300 y_i + \sum_{i=1}^{4}\sum_{j=1}^{4}(50+3(j-i))w_{ij}$$

s.t.

$$\sum_{i=1}^{4} w_{ij} = d_j \text{ for } j = 1, 2, 3, 4$$

$$w_{ij} \leq d_i y_i$$

$$d_1 = 200, d_2 = 300, d_3 = 100, d_4 = 400$$

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

$$w_{ij} \geq 0$$

Solving LP for both formulations, the first formulation yields $Z^1{}_{LP}$=50300. The second formulation yields $Z^2{}_{LP}$=51200. Therefore we conclude that the alternative formulation provides the tighter bound, hence is considered the superior formulation. Note that the global optimal solution for this example is found with the objective value (Z_{IP}) of 51200 under the lot for lot policy, where the production is planned each week to satisfy the demand of that week.

Other well-known reformulation approaches are Bender's reformulation and decomposition, column generation (also known as Dantzig-Wolf decomposition). See Bender (1962), Lemke and Spielbeg (1967), Magnanti and Wong (1981) for Benders' algorithm, and see Banhart et al. (1998), Desrochers et al. (1989), Parker and Ryan (1994), Savelsbergh (1997) for column generation.

2.2.3 Solution Techniques

Many problems can be effectively formulated as IPs. However, solving IP (i.e. to produce an optimal solution or to show that it is either unbounded or infeasible) is sometimes a very difficult task, especially as the number of variables increases. Tremendous efforts have therefore been put forward to develop effective IP solution techniques. Some well-accepted techniques are primal algorithms, branch and bound, Lagrangian relaxation and duality, decomposition, cutting planes, and heuristic method.

Nowadays, the IP solution process is usually done by employing IP software package. Most of the IP software package available today employ the LP based branch and bound algorithm at its core. Therefore, in the next section, it is very important to provide a discussion on this branch and bound technique.

2.2.4 LP based Branch and Bound

To find the best solution for an IP problem, one might in theory completely evaluate all feasible solutions in the solution space. However, this “naïve” approach is usually very time consuming and impractical for applying, especially with the large scale problems.

First proposed by Land and Doig in 1960 for linear programming, Branch and Bound (BB) and its variants has become the most commonly-used algorithm for solving IPs. BB is an intelligent enumeration strategy, which allows us to completely explore the solution space without having to evaluate all feasible solution. The use of bounds for the objective function combined with the value of the current best solution enables the algorithm to search parts of the solution space only implicitly (Clausen, 1999). The method employed in BB to find bound values can be used to classify type of BB, for instance, linear programming (LP) based BB, Lagrangian relaxation based BB, column generation based BB, and Lagrangian duality based BB. Note that different technique employed can result in dissimilar quality of bounds.

The idea behind BB is that we can partition the original problem, which might be difficult to solve, into the smaller subproblems, which are easier to solve, and solve them recursively. In general, BB consists of 3 key processes: subproblem selection, bounding, and branching. To find the optimal solution for the original IP problem, these processes might be repeating perform.

As earlier mention, most today available IP solver employed LP based BB algorithm, in this section we will start our talk on this type of BB. The general step of LP based B&B for a minimization problem can be outlined as follows:

Step 1: Subproblem selection

A subproblem is a problem derived from the original problem through addition of new constraints. Initially when we first start solving, there is only one original problem, which will be referred as root subproblem, to be selected to continue on step 2. However in later solving iteration, given a list of active subproblems, we will choose a subproblem to continue on step 2. There are a number of methods to decide which subproblem to continue. Some of the basic methods are left-to-right search, depth-first search, breadth-first search, and best-first search. Detail for each method can be found in Nemhauser and Wolsey (1988).

Step 2: Bounding

Bounds are used to determine if a subproblem is worth to continue exploring. To calculate a bound for any subproblem, we employ a relaxation technique. Solving a relaxation gives some idea on the best possible value of original problem (lower bound). For the minimization, the optimal objective value for relaxation is less than or equal to the optimal objective value for IP ($Z_{\mathrm{LP}} \leq Z_{\mathrm{LP}}$). In LP based BB, linear relaxation is used to derive bounds on the new subproblem. Linear relaxation transforms the original IP into a Linear Programming (LP) by relaxing the integrality restrictions. The results will be one of the following:

1) If the LP is infeasible, there is no solution for the subproblem.
2) If all integrality restrictions are satisfied, then we are done and the subproblem is solved. The objective value obtain from solving relaxation of the subproblem is an upper bound for all branches stemming from this subproblem.
3) If at least one of the integer variables is fractional in the LP solution, the solution obtained is a lower bound for the subproblem.

For the first two cases, we say the subproblem is inactive. For the third case, if the lower bound value of the subproblem exceeds the last updated best upper bound, we again say the subproblem is inactive. If a subproblem is inactive, pruning that subproblem. Otherwise the subproblem is active and will be continued on step 3 branching.

Step 3: Branching

To eliminate the fractional solution, new constraints are added to the subproblem. This process is called branching. We create new subproblems by branching on a fractional variable.

We continue the solving process by repeating step 1, 2 and 3 until all subproblems are inactive. At each round (or iteration), the best upper bound for all braches found so far is updated. And at the end of the solving process, the optimal solution of the IP is given by the current best upper bound.

To demonstrate the above BB steps, let us consider the very simple problem of: Minimize. It is also convenient to view BB steps in an enumeration tree.

Example 2-5

$Min -27x_1 - 28x_2 - 8x_3 - 9x_4$

s.t.

$7x_1 + 5x_2 + 4x_3 + 3x_4 \leq 17$

$6x_1 + 4x_2 + 8x_3 + 2x_4 \leq 9$

x_1 x_2 x_3 and x_4 are binary

Let us start the solving process with the first iteration.

Iteration 1

Step 1: Subproblem (or node) selection

Start with the LP relaxation of the original IP problem, referred as root subproblem (N_0). See in figure 2-4 the enumeration tree of the first iteration, the first node represents the node N_0.

Step 2: Bounding

Solve LP relaxation for the subproblem. $Z^0{}_{LP}$ = -50.5, X^0= (0.5, 1, 0, 1). We also know that the solution for the original problem $Z_{IP} \geq -50.5$

Step 3: Branching

We found that x_1 was not integer in our previous solution. Therefore we branch on x_1. To eliminate fractional solution, we create two new subproblems: subproblem 1 (N_1) is derived by adding x_1=0 to the root subproblem, subproblem 2 (N_2) is derived by adding x_1=1.

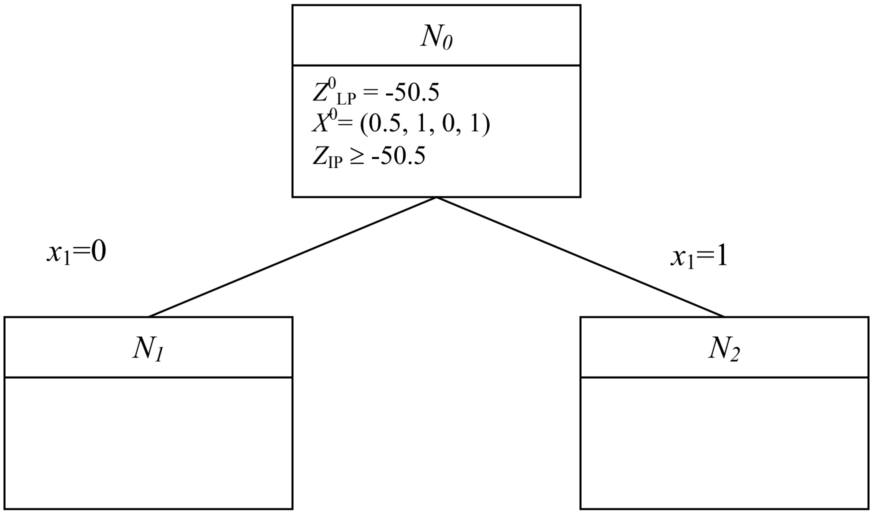

Figure 2-4. Graphical representation of example 2-5 Iteration 1

Iteration 2

Step 1: Subproblem selection

At this step, either N_1 or N_2 can be selected based on the selection technique. Let use the composite of depth-first search (DFS) and left-to-right search for this and the remaining iterations. Select N_1 to continue on step 2.

Step 2: Bounding

Solve LP relaxation for N_1. Z^1_{LP} = -40, X^1= (0, 1, 0.375, 1). N_1 is still active and need to continue branching. We also have new LB (tighter) for original IP where $Z_{\mathrm{IP}} \geq$ -40

Step 3: Branching

x_3 was not integer. We continue branching on x_3. To eliminate fractional solution, we create two new subproblems: subproblem 3 (N_3) is derived by adding x_3=0 to N_1, subproblem 4 (N_4) is derived by adding x_3=1.

Iteration 3

Step 1: Subproblem selection

We continue on a subproblem that is not inactive and again follow DFS and left to right search strategy. Select N_3 to continue on step 2.

Step 2: Bounding

Solve LP relaxation for N_3. $Z^3_{\mathrm{LP}} = -37$, $X^3 = (0, 1, 0, 1)$. Since all integrality restrictions are satisfied, we are done and the subproblem is solved. N_3 becomes inactive. The upper bound value (UB) for N_3 and all braches stemming from N_3 is -37. We also know that $-40 \le Z_{\mathrm{IP}} \le -37$. N_3 is pruned and no branching is needed. The best UB found so far becomes -37.

Iteration 4

Step 1: Subproblem selection

Select N_4 to continue on step 2

Step 2: Bounding

Solve LP relaxation for N_4. $Z^4_{\mathrm{LP}} = -15$, $X^4 = (0, 0.25, 1, 0)$. We compare the lower bound of this subproblem, which is Z^4_{LP}, to the last updated best UB, which is $Z^3_{\mathrm{LP}} = -37$. Since the lower bound exceed the last best updated UB, the subproblem is inactive and is pruned.

The BB enumeration tree for iteration 1 to 4 is shown in figure 2-5. As you can see from figure 2-4 and figure 2-5, the tree grows by branching process and shrinks if any subproblems are pruned. Now we continue solving process on the fifth iteration.

Iteration 5

Step 1: Subproblem selection

Select N_2 to continue on step 2.

Step 2: Bounding

Solve LP relaxation for N_2. $Z^2{}_{\mathrm{LP}} = -48$, $X^2 = (1, 0.75, 0, 0)$. We compare the lower bound of this subproblem, which is $Z^2{}_{\mathrm{LP}}$, to the last updated best UB, which is $Z^3{}_{\mathrm{LP}} = -37$. Since the lower bound does not exceed the last best updated UB, the subproblem is not inactive and need to continue branching.

Step 3: Branching

x_2 was not integer. We continue branching on x_2. To eliminate fractional solution, we create two new subproblems: subproblem 5 (N_5) is derived by adding x_2=0 to N_2, subproblem 6 (N_6) is derived by adding x_2=1.

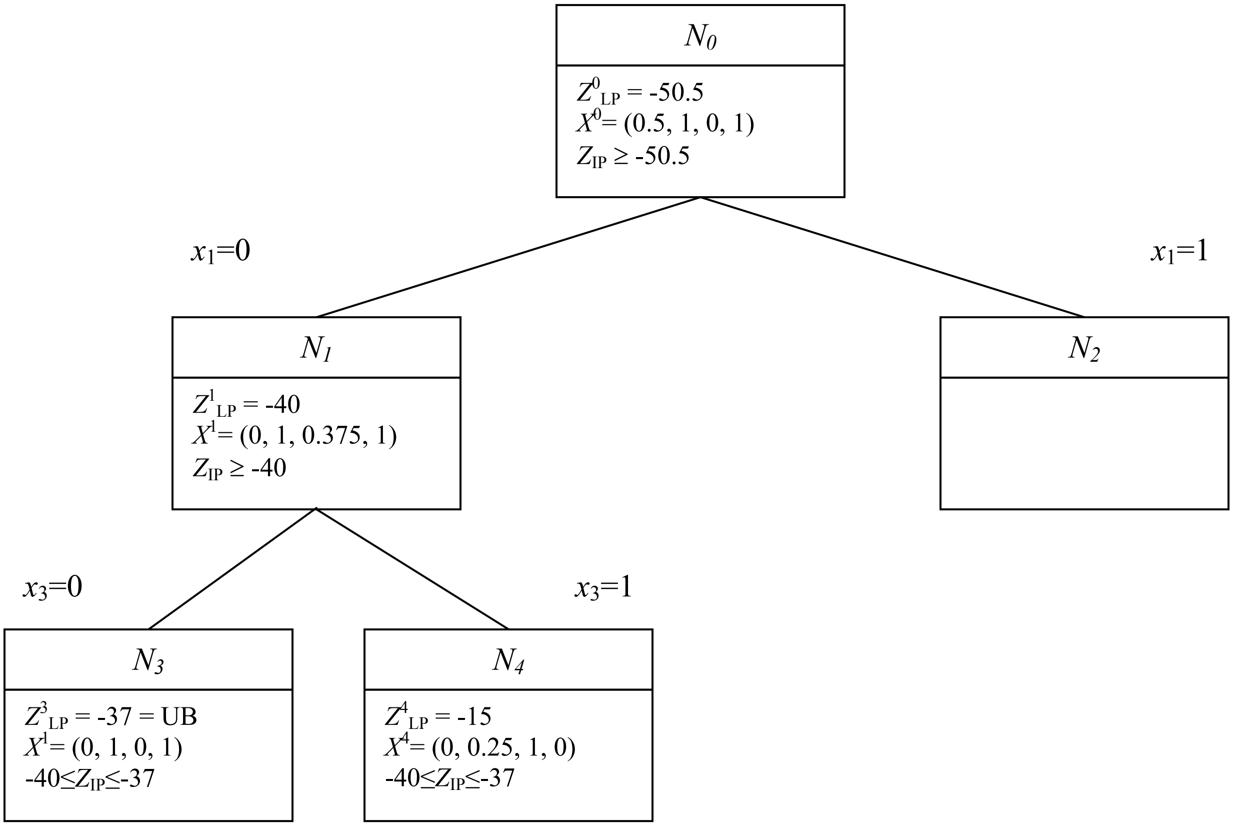
N_0
$Z^0{}_{\mathrm{LP}} = -50.5$
$X^0 = (0.5, 1, 0, 1)$
$Z_{\mathrm{IP}} \geq -50.5$
x_1=0
x_1=1
N_1
$Z^1{}_{\mathrm{LP}} = -40$
$X^1 = (0, 1, 0.375, 1)$
$Z_{\mathrm{IP}} \geq -40$
N_2
x_3=0
x_3=1
N_3
$Z^3{}_{\mathrm{LP}} = -37 = \mathrm{UB}$
$X^1 = (0, 1, 0, 1)$
$-40 \leq Z_{\mathrm{IP}} \leq -37$
N_4
$Z^4{}_{\mathrm{LP}} = -15$
$X^4 = (0, 0.25, 1, 0)$
$-40 \leq Z_{\mathrm{IP}} \leq -37$

Figure 2-5. Graphical representation of example 2-5 Iteration 5

Iteration 6

Step 1: Subproblem selection

Select N_5 to continue on step 2

Step 2: Bounding

Solve LP relaxation for N_5. $Z^5{}_{\mathrm{LP}} = -37$, $X^5= (1, 0, 0.125, 1)$. Since all integrality restrictions are still not satisfied, we compare $Z^6{}_{\mathrm{LP}}$ to the last updated best UB, which is $Z^3{}_{\mathrm{LP}} = -37$. N_5 is still active and need to continue on branching step since $Z^5{}_{\mathrm{LP}}$ does not exceed -37.

Step 3: Branching

x_3 was not integer. We continue branching on x_3. To eliminate fractional solution, we create two new subproblems: subproblem 7 (N_7) is derived by adding x_3=0 to N_5, subproblem 8 (N_8) is derived by adding x_3=1.

Iteration 7

Step 1: Subproblem selection

Select N_7 to continue on step 2

Step 2: Bounding

Solve LP relaxation for N_7. $Z^7{}_{\mathrm{LP}} = -36$, $X^7= (1, 0, 0, 1)$. Again all integrality restrictions are satisfied, we are done and the subproblem is solved. The upper bound value (UB) for N_7 and all braches stemming from N_7 is -36. However the UB of node N_7 is not better than that of N_3. The best UB found so far still remains -36. Since N_7 becomes inactive, the node is pruned.

Iteration 8

Step 1: Subproblem selection

Select N_8 to continue on step 2

Step 2: Bounding

Solve LP relaxation for N_8. The LP solution is infeasible; we conclude that IP for N_8 is also infeasible. N_8 becomes inactive and is pruned.

Iteration 9

Step 1: Subproblem selection

Select N_6 to continue on step 2

Step 2: Bounding

Solve LP relaxation for N_6. The LP solution is infeasible; we conclude that IP for N_6 is also infeasible. N_6 becomes inactive and is pruned. There is no more active subproblem to continue branching. The LP based BB process is terminated. The optimal solution of the original IP is given by the current best upper bound, which is equal to $Z^3{}_{\mathrm{LP}}$ = -37.

The complete enumeration tree for example 2-5 is shown in figure 2-6.

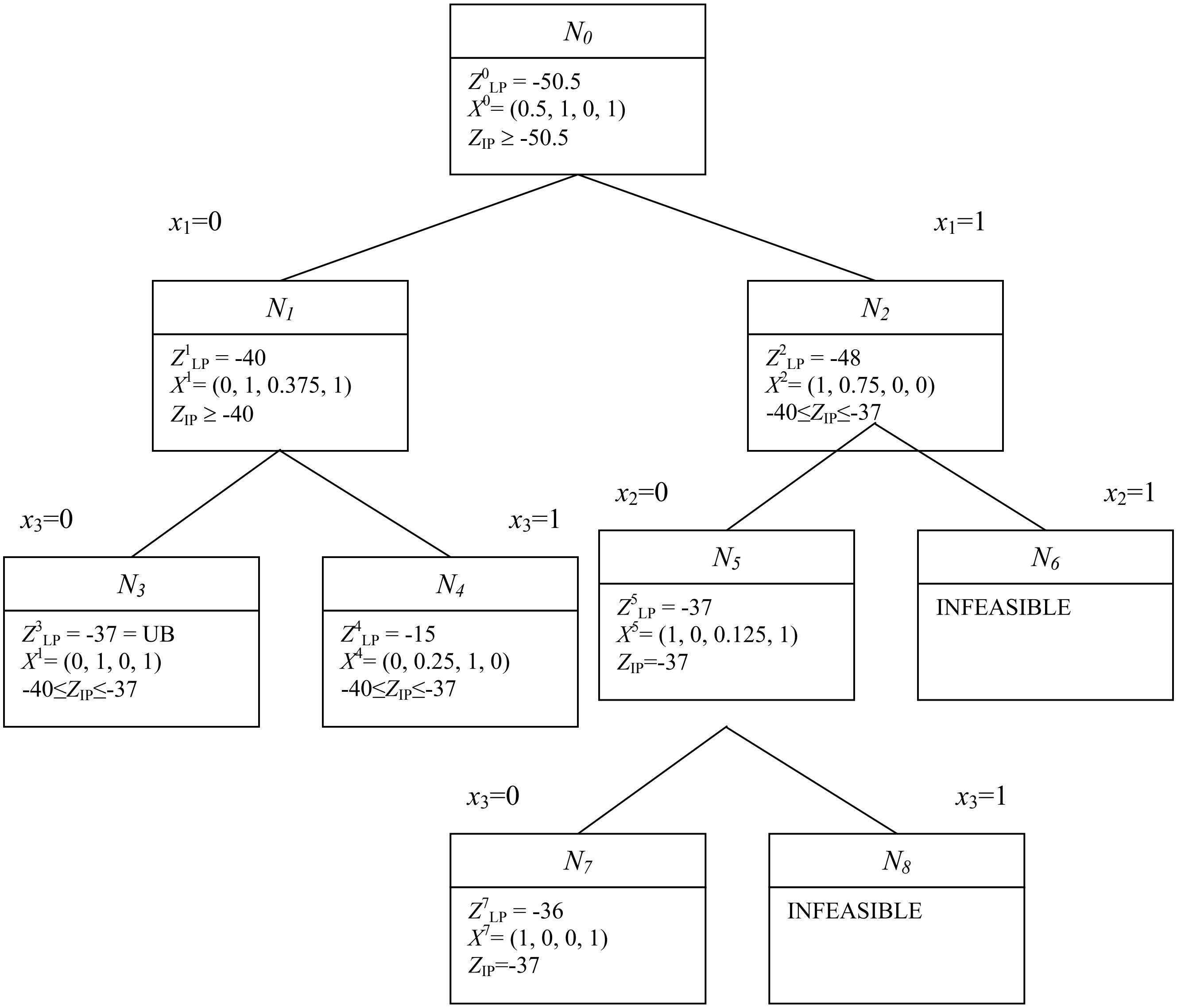

Figure 2-6. Graphical representation of example 2-5 Iteration 9

2.2.5 Generation of LP based Branch and Bound

LP based BB is a very general approach. It is a standard way of dealing with IP. Despite of its easiness to solve, this method often seems to be inefficient since the bounds generated by linear relaxation are weak. Tremendous progress has been made over the years to improve LP bounds and make LP based BB more attractive. One direction of the improving is to develop composite procedures by combining LP based BB with the other solution techniques such as cutting plane algorithm, the more efficient relaxation approaches, and the other bound tightening techniques.

Integration between BB and cutting plane methods has been in great interest since it could potentially lead to a considerable reduction in number of iteration for solving an IP (see Mitchell, 2000; Nemhauser and Wolsey, 1988; and references therein for a good introduction). The key idea is to strengthen bond values by the additional of cutting planes at one or more subproblems along the BB process. If cut(s) is adding only at the root subproblem, the technique is called Cut and Branch (CB) technique, which is one of the reformulation technique explained in section 2.2.2. If cut(s) is adding at any other nodes (not the root node) after branching step, it is referred as Branch and Cut (BC).

Branch and Cut (BC) is an exact approach for solving IP. It has been in great interest since the development of polyhedral theory and the introduction of strong problem specific cutting planes (Mitchell, 2002). While adding more cuts makes the subproblem becomes bigger and consequently takes longer time to solve, adding the “good” cuts can improve the formulation and tighten bounds, which in turn significantly improve the BB’s solving efficiency. The BC basic processes of each iteration include subproblem selection, bounding, branching, and adding cuts where needed.

Here, we use the following two examples to explain the CB and BC methods. Let revisit example 2-5. Instead of solving by BB, we will use CB for example 2-6 and BC for example 2-7 to solve the problem.

Example 2-6

$Min -27x_1 - 28x_2 - 8x_3 - 9x_4$

s.t.

$7x_1 + 5x_2 + 4x_3 + 3x_4 \leq 17$ (N_0)

$6x_1 + 4x_2 + 8x_3 + 2x_4 \leq 9$

x_1 x_2 x_3 and x_4 are binary

To perform CB, we first reformulate the original IP by finding cuts to improve the formulation. As mentioned earlier, there are server methods to find cuts. However, in this example, a cover inequality for knapsack polytope is used. The detail on this method can be found in Nemhauser and Wolsey (1988). We obtain $x_1 + x_2 \leq 1$ (cut 1) from the second constraint as a good cut for the root subproblem (N_0). After appending cut 1 into the original IP, we solve the LP of N'_1.

$Min -27x_1 - 28x_2 - 8x_3 - 9x_4$

s.t.

$7x_1 + 5x_2 + 4x_3 + 3x_4 \leq 17$ (N'_0) → N_0 + cut 1

$6x_1 + 4x_2 + 8x_3 + 2x_4 \leq 9$

x_1 x_2 x_3 and x_4 are binary

$x_1 + x_2 \leq 1$

Iteration 1

Step 1: Subproblem selection

Select N'_0 to continue on step 2

Step 2: Bounding

Solve LP relaxation for N'_0. $Z^{0\prime}{}_{LP}$ = -40, $X^{0\prime}$= (0, 1, 0.375, 1). We also know that the solution for the original problem $Z_{IP} \geq$ -40

Step 3: Branching

We found that x_3 was not integer in our previous solution. Therefore we branch on x_3. To eliminate fractional solution, we create two new subproblems: subproblem 1 (N_1) is derived by adding x_3=0 to the root subproblem, subproblem 2 (N_2) is derived by adding x_3=1.

Iteration 2

Step 1: Subproblem selection

Select N_1 to continue on step 2

Step 2: Bounding

Solve LP relaxation for N_1. Z^1_{LP} = -37, X^1= (0, 1, 0, 1). Since all integrality restrictions are satisfied, we are done and the subproblem is solved. N_1 becomes inactive. The upper bound value (UB) for N_1 and all braches stemming from N_1 is -37. We also know that $-40 \leq Z_{\mathrm{IP}} \leq -37$. N_1 is pruned and no branching is needed. The best UB found so far becomes -37.

Iteration 3

Step 1: Subproblem selection

Backtracking on the path from N_1 toward the root, we find N_2 to continue on step 2

Step 2: Bounding

Solve LP relaxation for N_2. Z^2_{LP} = -15, X^2= (0, 0.25, 1, 0). We compare the lower bound of this subproblem, which is Z^2_{LP}, to the last updated best UB, which is Z^1_{LP} = -37. We found that the lower bound exceed the last best updated UB, the subproblem is inactive and is pruned. There is also no need to update the best UB found so far. Since there is no more active subproblem to continue branching, the LP based CB process is terminated. The optimal solution of the original IP is given by the current best upper bound, which is equal to Z^1_{LP} = -37. The CB enumeration tree for this example is shown in figure 2-7. Note that if we can find a strong cut to add to the root subproblem, we might be able to obtain the optimal solution from solving LP of the reformulate subproblem.

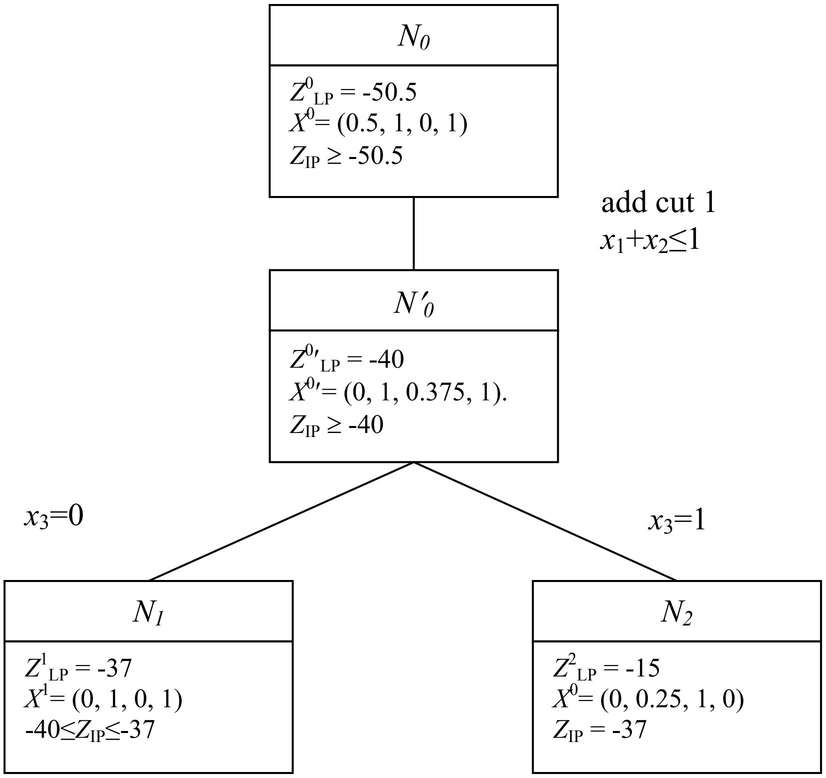

Figure 2-7. Graphical representation of example 2-6 Iteration 3

Instead of using cover inequality method to find cut, let us use a fractional cutting-plane algorithm (FCPA). The detail of FCPA can be found in Nemhauser and Wolsey (1988). We obtain $x_1 + x_2 + x_3 \leq 1$ (cut 2) as a good cut for the root subproblem (N_0). After appending cut 1 into the original IP, we solve the LP of N'_0.

$Min - 27x_1 - 28x_2 - 8x_3 - 9x_4$

s.t.

$7x_1 + 5x_2 + 4x_3 + 3x_4 \leq 17$ (N'_0) → N_0 + cut 2

$6x_1 + 4x_2 + 8x_3 + 2x_4 \leq 9$

x_1 x_2 x_3 and x_4 are binary

$x_1 + x_2 + x_3 \leq 1$

Solving LP of N'_0, we obtain the optimal solution $Z^0{}_{LP}$ = -37, X^0= (0, 1, 0, 1).

Example 2-7

We solve the IP problem in example 2-5 again with BC.

For the sake of illustration, let us follow the same steps as in example 2-5 for iteration 1.

- In iteration 1, we found $Z^0{}_{\mathrm{LP}}$ = -50.5, X^0= (0.5, 1, 0, 1). Therefore we created two new subproblems: N_1 is derived by adding x_1=0 to N_0, and N_2 is derived by adding x_1=1 to N_0.

Let us now start employing the cutting plan method at iteration 1 after branching.

Iteration 1 (continue)

Step 4: Adding cuts (apply FCPA to find cuts)

Before start adding cuts we solve LPs of N_1 and N_2 to indicate if cuts are needed. If the LP solution of a node is satisfied all integer restriction, we don't need to add cut since the problem has been solved. Let us look back to our example 2.6, we found that $Z^1{}_{\mathrm{LP}}$ = -40, X^1= (0, 1, 0.375, 1) and $Z^2{}_{\mathrm{LP}}$ = -48, X^2= (1, 0.75, 0, 0). Therefore cuts can be added to both nodes to tighten bounds. We have $3x_1 + 4x_2 + 4x_3 + 4x_4 \le 8$ (cut 1) as a good cut for subproblem N_1 and $3x_1 + 2x_2 + 4x_3 \le 3$ (cut 2) as a good cut for subproblem N_2. Note that cut 1 is a valid inequality for N_1 and its associated children nodes and cut 2 is a valid inequality for N_2, and its associated children nodes. But both cuts need not to be valid for N_0. After append the cuts, we generate two new subproblems: N'_1 and N'_2.

$Min - 27x_1 - 28x_2 - 8x_3 - 9x_4$

s.t.

$7x_1 + 5x_2 + 4x_3 + 3x_4 \le 17$ (N'_1) → N_1 + cut 1

$6x_1 + 4x_2 + 8x_3 + 2x_4 \le 9$

$x_1 = 0$

$3x_1 + 4x_2 + 4x_3 + 4x_4 \le 8$

x_1 x_2 x_3 and x_4 are binary

and

$Min - 27x_1 - 28x_2 - 8x_3 - 9x_4$

s.t.

$7x_1 + 5x_2 + 4x_3 + 3x_4 \le 17$ (N'_2) → N_2 + cut 2

$6x_1 + 4x_2 + 8x_3 + 2x_4 \le 9$

$x_1 = 1$

$3x_1 + 2x_2 + 4x_3 \le 3$

x_1 x_2 x_3 and x_4 are binary

Iteration 2

Step 1: Subproblem selection

Select N'_1 to continue on step 2

Step 2: Bounding

Solve LP relaxation for N'_1. $Z^{1\prime}_{\mathrm{LP}} = -37$, $X^{1\prime} = (0, 1, 0, 1)$. Since all integrality restrictions are satisfied, we are done and the subproblem is solved. N'_1 becomes inactive. The upper bound value (UB) for N'_1 and all braches stemming from N'_1 is -37. We also know that $-50.5 \le Z_{\mathrm{IP}} \le -37$. N'_1 is pruned and no branching is needed. The best UB found so far becomes -37.

Iteration 3

Step 1: Subproblem selection

Backtracking on the path from N'_1 toward the root, we find N'_2 to continue on step 2

Step 2: Bounding

Solve LP relaxation for N'_2. $Z^{2\prime}_{\mathrm{LP}} = -36$, $X^{2\prime} = (1, 0, 0, 1)$. Again here the subproblem is solved as x_1, x_2 and x_3 are all integer. N'_2 becomes inactive. There is no need to continue branching further. We compare the upper bound of this subproblem, which is $Z^{2\prime}_{\mathrm{LP}}$, to the last updated best UB, which is $Z^{1\prime}_{\mathrm{LP}} = -37$. We found that $Z^{1\prime}_{\mathrm{LP}} \le Z^{2\prime}_{\mathrm{LP.}}$ Therefore we no need to update the best UB found so far. Since there is no more active subproblem to continue branching, the LP based BC process is terminated. The optimal solution of the original IP is given by the current best upper bound, which is equal to $Z^{1\prime}_{\mathrm{LP}} = -37$.

Complete enumeration tree for example 2-7 is shown in figure 2-8.

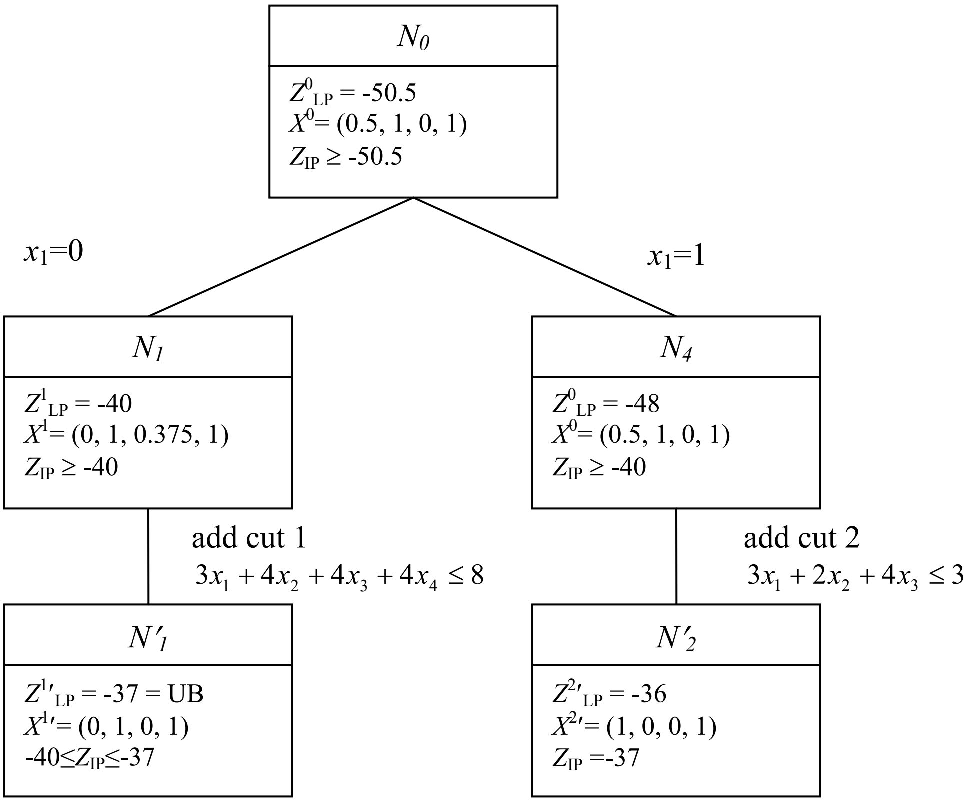

Figure 2-8. Graphical representation of example 2-7 Iteration 3

As clearly illustrated by example 2-5, 2-6 and 2-7, with both CB and BC, we can bring down number of solving iteration. From the examples, we have shown that adding cutting plan can substantially reduce the size of BB enumeration tree and in turn result in the reduction of the overall computation time.

By combining the more efficient relaxation approaches to the LP based BB, we expect either the strong bound value to be generated so that more nodes can be pruned or the easier to solve subproblem, which will consequently speeds up the LP based BB process. A relaxation, other than the LP relaxation, can be obtained 1) by dropping the complicating constraints of the original IP problem and adding them into the objective function with the penalty term if they are not satisfied 2) by adding and/or changing variables. The former method is known as Lagrangian relaxation (LR) method, and the later is known as combinatorial relaxation method. The review of the both relaxation methods can be found in

Nemhauser and Wolsey (1988). Example 2-8 bellow illustrates how the integration approaches between BB and the LR method can possibly improve the basic LP based BB.

Example 2-8

In this example LR technique is embedded into the basic LP based BB and used to solve the IP problem in example 2-5. Let follow the same steps as in example 2-5 for iteration 1 to 4.

- In iteration 4, we found Z^4_{LP} = -15, X^4= (0, 0.25, 1, 0). The last updated best UB is Z^3_{LP} = -37.

Iteration 5

Step 1: Subproblem selection

Backtracking on the path from N_4 toward the root, we find N_2 to continue on step 2.

Step 2: Bounding

Let us now start employing the LR technique to subproblem N_2. To demonstrate the concepts of apply LR to LP based BB, the constraint "$7x_1 + 5x_2 + 4x_3 + 3x_4 \le 17$" is randomly selected to be relaxed since this example does not contain the complicating constraints.

The Lagrangian relaxation of N_2 with respect to $7x_1+5x_2+4x_3+3x_4 \le 17$ is

$Z_{LR}(\lambda) = Z_{LR}(\lambda) = Min - 27x_1 - 28x_2 - 8x_3 - 9x_4 + \lambda(7x_1 + 5x_2 + 4x_3 + 3x_4 - 17)$

s.t.

$6x_1 + 4x_2 + 8x_3 + 2x_4 \le 9$ (N'_2) → LR of N_2

$x_1 = 1$

x_1 x_2 x_3 and x_4 are binary

Solving the subproblem N'_2 with λ=0, we obtain $Z^{2\prime}_{LR}$ (λ=0) = -36, $X^{2\prime}$ = (1, 0, 0, 1). Again all integrality restrictions are satisfied, we are done and the subproblem is solved. N'_2 becomes inactive and no branching is needed. The UB found from solving LR is -36, which exceeds the last best updated UB, the subproblem is inactive and is pruned. The BB process is terminated. The optimal solution of the original IP is given by the current best upper bound, which is equal to $Z^{4}_{\text{LP}} = -37$.

The enumeration tree for example 2-8 is shown in figure 2-9. Although the LR method does not guarantee the optimal solution of the relaxed subproblem, finding a good LR solution might still improve the solving efficiency. Note that the reason for solving LR here is not necessarily to get the optimal solution but to obtain good lower bounds which can increase the possibility of pruning the nodes. However, a potential effective way to obtain the optimal solution is to integrate duality, in conjunction to the LR into the LP based BB. The method is knows as Lagrangian duality (LD).

Another variation of LP based BB, which allows column generation to be applied throughout the BB enumeration tree, is called branch and price (BP). The key idea of BP is that we can easy the solving process of the LP relaxation, or subproblem, by leaving out some of the decision variables (also referred as columns), that in most case will equal to zero at the optimal solution anyway. To check the optimality of the subproblem, the pricing step is performed. The dual of the LP (also referred as pricing problem) is constructed and solved to identify if any decision variable should be added. If such variables are found, we add them into the LP and reoptimize the problem. If the optimal solution of the subproblem satisfied the integer restriction, we are done with the subproblem. Otherwise, continue with branching on a fractional variable. The general idea of BP is quite similar to that of BC. Except, in BP the cuts are added (also known as row generation) throughout the BB enumeration tree to tighten bound, while BC focuses on adding more decision variables (also known as column generation). BP is a good approach for problems that have many variables or columns to handle efficiently, but relatively few constraints such as assignment problem, routing problem and scheduling problem. An excellent introduction for BP can be found in Barnhart et al. (1998).

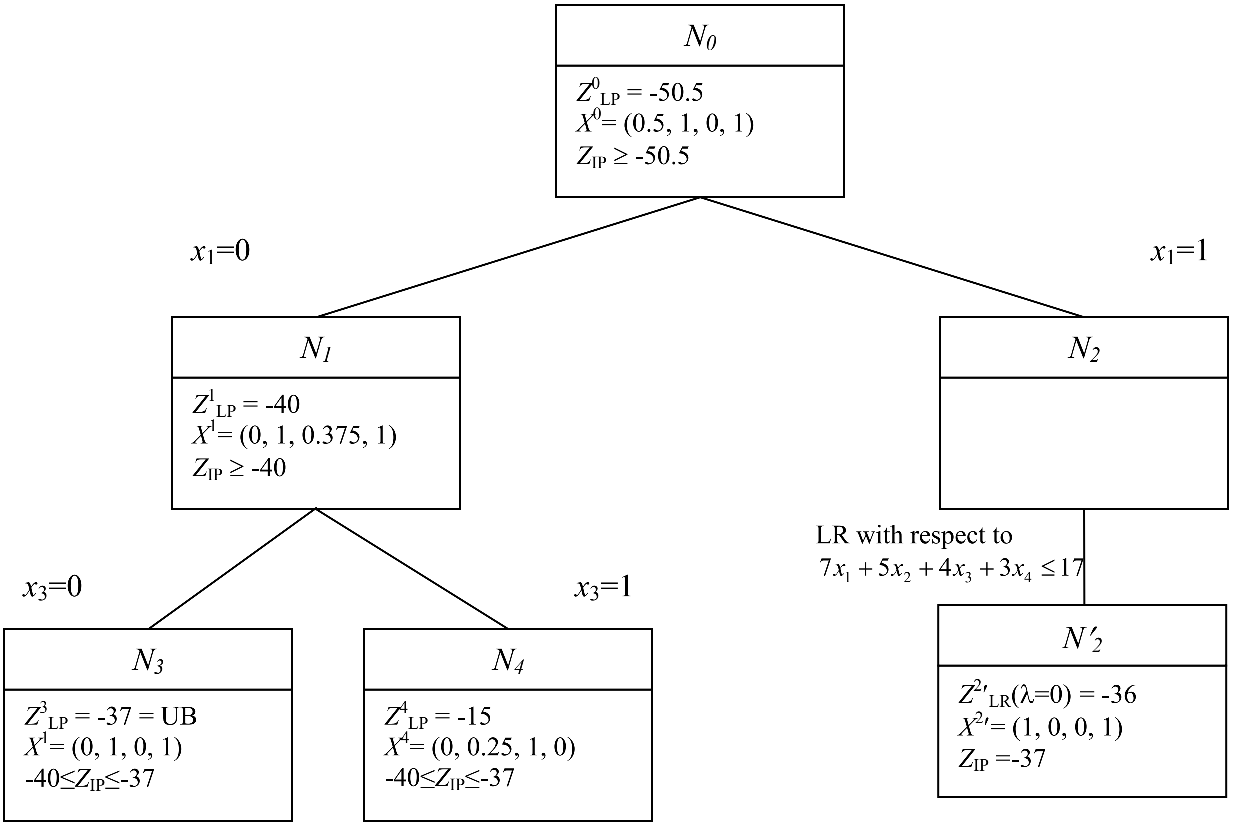

Figure 2-9. Graphical representation of example 2-8 Iteration 5

While integrating cutting plane algorithm and/or the more efficient relaxation approaches attempts to improve the lower bound for minimization, primal heuristics can be employed to obtain the tighter upper bound. The idea of applying primal heuristics to the LP based BB is using simple and quick heuristic either to obtain a good feasible solution for the IP or to find a good upper bound and use the solution to increase the nodes pruning possibility. The primal heuristics can be general methods (such as fixing some variable value and use optimization to find the value of the remaining variables, simple round up or down from the LP solution, greedy algorithm, and k-interchange heuristic) or problem specific methods (such as first come first served (FCFS) for job sequencing problem, short processing time (SPT) for production scheduling, Hungarian algorithm for assignment problem).

Combining more than one technique to LP based BB can also be an alternative. The benefit will depend on how fast the problem can be solved and how good the solution is.

Generally, there always is the trade off between solution time and quality of the solution. However, regardless of which techniques are engaged, the tighter bound provided by the composite approaches might be of great help in decreasing the solution time of LP based BB.

CHAPTER 3. SCHEDULING INBOUND CALLS IN CALL CENTERS

The characteristics of a basic inbound call center are described in Chapter 1. As previously mentioned, the method of matching calls with the proper CSRs is vital to providing efficient service, and hence becomes a focus of our research study.

In this chapter, we first introduce three IP models as instruments to handle inbound calls scheduling task. We also examine the effectiveness of the models under various call centers environments through the numerical result. Finally, we present some conclusion that can be made based on these models.

3.1 Model Development

As noted in the previous in Chapter 2, inbound call scheduling has not been addressed previously as an optimization problem, and in this section we formulate this task as an IP problem. We start by describing the system and the modeling assumption. A call center has a known number of different classes of CSRs and a given number of different call types. At a particular point in time, there is a known number of different types of calls either being processed by the CSRs or waiting to be processed. Each class of CSRs has its own skill set, which means that the CSRs in that class are capable of handling a given type of calls at a specific service time.

Each class of CSR has a corresponding queue of calls waiting to be processed, and when a call comes in it is assigned to one of those queues by an Automatic Call Distributor (ACD). The calls waiting in each queue can also be sequenced based on their characteristics and priority. The scheduling model addresses how at any point in time the ACD should make these assignment and sequencing decisions given the set of calls in the system. For simplicity, we assume that there are no lost calls at any time, and that when a call is taken by a CSR it will be served without pre-emption at a server-dependent serving time. Furthermore, a call will immediately leave the system after being served.

Specifically, we assume that at a given time t, a finite set I of $m(t)$ calls is waiting to be scheduled by ACD. Each call has its own waiting time $w_i(t)$ attached, which may depend on the type of call as well as other factors. Calls are sequenced and distributed to one of parallel queues of a given set J of $n(t)$ CSRs. Note that each call can be only processed once and only by one CSR, who may have an earliest time that he/she can take a next call, which we refer to as the remaining serving time $r_j(t)$. Thus, the CSR assignment and call sequencing can be though of jointly as assigning each call to one of $m(t) \times n(t)$ possible positions.

The optimization model aims to identify the best call schedule, that is, the assigned CSR for each call, and position of the call in queue of that CSR. The decision variables for the optimization problem can thus be defined as follows:

$$X_{ijq}(t) = \begin{cases} 1 & \text{if call } i \text{ is assigned to CSR } j \text{ at queue position } q, i \in I, j \in J, q \in Q, \\ 0 & \text{otherwise.} \end{cases}$$

Most call centers appear to use a target customer service level as the primary performance measure. One of the common ways to do so is to guarantee a minimum service level, which is usually defined as some fixed Z percent of calls answered in Y seconds. Certain industry standards exist for these values, for example the popular 80/20 rule of Z = 80, Y = 20 (Koole and Mandelbaum, 2002). Another desirable property is workload balance. If the focus is solely on the service level, it is possible for some CSRs to be idle or waiting for work while other CSRs are overworked. In such situations, call center managers may be interested in balancing work load while maintaining the service level. The primary motivation for this is the underlying assumption that when CSRs do not experience large fluctuations in call traffic they are likely to be more productive throughout the day.

While guaranteeing certain service level is the most common performance measure used by call center managers, from an optimization perspective this may be better represented as a constraint than an objective function. We therefore consider two types of objective function that are believed to be related to achieving good service level, namely minimizing the flow time of the calls and balancing the load of the CSR.

Specifically, we consider two flow time related objectives: minimizing the total flow time and minimizing the maximum flow time for any call. For balancing the load we choose to minimize the maximum deviation of assigned work load to guarantee that no single CSR is assigned unusually high workload.

The following variables will be convenient in formulating the optimization models. First, we define the queue position of call *i* served by CSR as

$$Q_i(t,X) = \sum_{q=1}^{m(t)} qX_{iJ_i(t,X)q}(t) \quad , \tag{3.1}$$

where J_i (*t,X*) is the CSR which is assigned at time *t* to serve call *i*, that is,

$$J_i(t,X) = \sum_{q=1}^{m(t)} \sum_{j=1}^{n(t)} jX_{ijq}(t) \, . \tag{3.2}$$

Now given the queue position, assignment, and the server dependent processing time P_{ij}, queue length is the total processing time spent by a CSR to complete calls prior than call *i* in a queue and can be calculated at time *t* as follows:

$$QL_i(t,X) = \sum_{q=1}^{Q_i(t)-1} \sum_{i=1}^{m(t)} P_{ij} X_{iJ_i(t,X)q}(t) \, . \tag{3.3}$$

Furthermore, the speed answer or the total time it takes to answer call *i* is then given by

$$SA_i(t,X) = w_i(t) + \sum_{q=1}^{m(t)} \sum_{j=1}^{n(t)} [r_j(t) X_{ijq}(t)] + QL_i(t,X) \, . \tag{3.4}$$

The cumulative assigned work load of CSR *j* calculated at time *t* can be calculated as follows:

$$Cl_j(t,X) = ACl_j(t_0) + \sum_{q=1}^{m(t)} \sum_{i=1}^{m(t)} P_{ij} X_{ijq}(t) \, , \tag{3.5}$$

where $ACl_j(t_0)$ is the actual cumulative workload at time t_0, and the actual cumulative work load of CSR *j* calculated at time *t* is

$$ACl_j(t,X) = ACl_j(t_0) + \sum_{q=1}^{s_j(t,X)} \sum_{i=1}^{m(t)} P_{ij} X_{ijq}(t) \, . \tag{3.6}$$

Finally, the number of calls that CSR *j* started the service after time *t* and before $t_{1,}$ $s_j(t,X)$ satisfies

$$s_j(t,X) = \max\left\{ s_j(t,X) : r_j(t) + \sum_{q=1}^{s_j(t,X)-1} \sum_{i=1}^{m(t)} P_{ij} X_{ijq}(t) \le t_1 - t, s_j(t,X) \in Z^+ \right\}. \quad (3.7)$$

With this notation in hand, we can now formulate the first mathematical programming problem for assigning calls with the objective of minimizing the total flow time (TFT).

Problem TFT

$$\min_{\mathbf{X}} \sum_{i=1}^{m} F_i(t,X) = \sum_{i=1}^{m} \left(SA_i(t,X) + \sum_{q=1}^{m(t)} \sum_{j=1}^{n(t)} P_{ij} X_{ijq}(t) \right), \quad (3.8)$$

Subject to

$$\sum_{q=1}^{m(t)} \sum_{j=1}^{n(t)} X_{ijq}(t) = 1, \forall i, \quad (3.9)$$

$$\sum_{i=1}^{m(t)} X_{ijq}(t) \le 1, \forall j, q. \quad (3.10)$$

Constraint (3.9) ensures that each call is served exactly once by exactly one CSR. Constraint (3.10) ensures that not more than one call can be served at a time by a CSR.

Sometimes minimizing the total flow time might lead to an unbalanced solution where certain calls take very long to process. To address this, we also consider the alternative objective function of minimizing the maximum flow time (MFT), subject to the same constraints as before.

Problem MFT

$$\min_{\mathbf{X}} \max_{i} \{F_i(t,X)\} = \max_{i} \left\{ SA_i(t,X) + \sum_{q=1}^{m(t)} \sum_{j=1}^{n(t)} P_{ij} X_{ijq}(t) \right\} \tag{3.11}$$

Subject to constraints (3.9) to (3.10).

Finally, as mentioned above, minimizing any flow time related objective may result in loss of balance between work assignments to different CSRs. Thus, we also consider the problem where the objective is to minimize the maximum deviation of cumulative assigned workload (MDCAW).

Problem MDCAW

$$\min_{\mathbf{x}} \max_{j} \left| Cl_j(t,X) - \frac{1}{n(t)} \sum_{j=1}^{n(t)} Cl_j(t,X) \right|, \tag{3.12}$$

Subject to constraints (3.9) and (3.10).

As earlier mentioned, many call centers promise a given customer service and it may therefore be appropriate to formulate a service level constraint as follows:

$$\frac{100}{m(t)} \sum_{i=1}^{m(t)} \chi_{\{SA_i(t,X) \le Y\}} \ge Z \text{ , } m(t) > 0 \tag{3.13}$$

where $\chi_{\{SA_i(t,X) \le Y\}} = \begin{cases} 1 & \text{if } SA_i(t,X) \le Y \\ 0 & \text{otherwise} \end{cases}$.

This constraint can be used with any of the three IPs described above. Finally, for the flow time related objectives, a constraint may be added to manage the deviation of the cumulative assigned work load of any CSR. Similar to the service level Z in equation (3.13), this can be done by specifying a load balance factor L and adding the following constraint:

$$\left| Cl_j(t,X) - \frac{1}{n(t)} \sum_{j=1}^{n(t)} Cl_j(t,X) \right| \leq L \text{ , } n(t) > 0. \qquad (3.14)$$

Note that the reoptimize process is required if there is any call coming or leaving the system. In the next section we explore the call center performance improvements that can be obtained through the use of these IP formulations.

3.2 Numerical Example

This section describes the experimental design we have conducted and also reports the results. To investigate the effectiveness of the developed optimization models, a realistic call system was simulated and investigated under various scenarios that correspond to different call center characteristics and management preferences. In addition to the three objective functions used by the optimization problems, that is, total flow time, maximum flow time, and maximum deviation of cumulative assigned workload, other performance measures relevant to a call center are evaluated. This includes the average number of calls in the system, average call flow time, average call waiting time, the CSR work load, and the service level of the call center. These various performance measures are analyzed and the performance used to determine which method for calls scheduling should be used in which circumstance. The details of the experimental design are described next.

3.2.1 Experimental Design

The call center used for this study is modeled based on an actual operational call center with the parameters modified to be achieve the experimental goals. In this model system, there are 43 CSRs working under 8 supervisors. Each CSR is scheduled to work 7.5 hours per day excluding break and lunch time. Shifts are planned to meet the incoming call volumes. Only two or three CSRs will share the same schedule in order to spread the available work evenly throughout the center's working hours from 8AM to 6PM. CSRs are skilled to work on multiple types of calls. Most CSRs are specialized in certain types of calls, and hence have faster service rates for those calls, while only minimally trained on others, and hence have slower service rates on those calls. Although this model call center can handle both inbound and outbound calls only inbound calls are treated in this study.

The experiment is designed to be realistic based on the model provided by the actual call center described above. The parameters that define the call center include the utilization rate of the CSRs, the types of calls that arrive, the CSR skill distribution, and the time horizon considered. Calls and CSR attributes' information in the call center are tracked and recorded by the contact management system (CMS). The CMS allows going back and tracking daily reports by a 30-minute period for each individual CSR. Thus, all the experiments performed here were performed for 30-minute time intervals. This is indeed one of the most common time intervals using for summarizing call-by-call data (Koole and Mandelbaum, 2002), with other practical time intervals being 15-minute and 60-minute period (Gans et al., 2003).

Generally the numbers of CSRs and the numbers of incoming calls for an inbound call center fluctuate widely during the day. The proposed models therefore allow the consideration of any numbers of CSRs and calls. However, given a relatively short time horizon, such as the 30-minute intervals considered in this experiment, the number of CSR can be assumed to be fixed. In the actual call center used as a model, two or three CSR share the same work schedule and would work the same 30-minute periods. Thus, in our experiments the number of CSRs in the call system was assumed to be three for all experiments while the number of incoming calls may vary based on the utilization rate of the call center.

According to Feinberg (1990), the utilization rate of the agents is the major factor that affects the performance characteristics of ACD system. For instance, the major performance characteristics of the system are similar whether the simulation of the system with 400 new clients per hour (on average) and 20 agents or with 4000 new clients per hour (on average) and 200 agents. In a large best-practice call center, it is found that the utilization levels of the agents can average between 90 and 95 percent (Mandelbaum, 2003). Consistent with these observations and the actual utilization rates in our model call center, we vary the utilization rate of the call center in our experiments from low (85%), to intermediate (90%), and high (95%).

Most call centers must process several types of calls and as noted earlier CSRs may be specialized to better handle certain types. For this experiment, we assume that there are three types of calls, which we simply refer to as Type I, Type II, and Type III, with approximately one half of the call being Type I, approximately one third Type II, and approximately one sixth Type III. We can thus start with the desired utilization and calculate the arrival rate for each type so that the total arrival rate satisfies this utilization and the approximate proportions of each call type are preserved (see Table 3.1). To generate the actual call arrivals from these rates, it is assumed that the time between call arrivals is exponentially distributed, which is a reasonable assumption for many call centers.

Table 3.1: Arrival Rates for Each Type of Call

	Utilization Rate		
	85%	*90%*	*95%*
Calls' arrival rate: λ (calls/hr)	54	57	60
Type I call's arrival rate: λ_1 (calls/hr)	28	29	31
Type II call's arrival rate: λ_2 (calls/hr)	19	21	22
Type III call's arrival rate: λ_3 (calls/hr)	7	7	8

In most call centers, including the model call center used to design these experiments, at least some of the CSRs are specialized in certain types of calls but also at least minimally trained in other types of calls. In our experiment we consider this case but also the two extreme cases of no specialization and extreme specialization where there is no training for other calls. Thus, three cases are considered in the experiment: In Case I, each CSR is skilled to work with only one type of call, that is, each CSR is specialized with no training for other calls. In Case II, there are three identical CSRs that skilled to work with any types of call, that is, there is no specialization. In Case III, the three CSRs are skilled to work with any types of call but at the different service rate, that is, they are specialize in one type of call but also trained to handle the other two types. The average service times for each of these three cases are summarized in Table 3.2. To generate the actual service time for each time it is assumed that service times are exponentially distributed, which is often a good approximation to the service time distribution (Feinberg, 1990).

Table 3.2: Service Time for Different CSRs for Each Type of Call

Service Time Case	*Type of call*	*CSR 1*	*CSR 2*	*CSR 3*
Case I	Type I	147	-	-
	Type II	-	199	-
	Type III	-	-	175
Case II	Type I	147	147	147
	Type II	199	199	199
	Type III	175	175	175
Case III	Type I	167	80	233
	Type II	394	118	198
	Type III	198	122	54

Given the parameter settings described above, the experiment considers using optimization with each of the three mathematical programming models (that is, TFT, MTF, and MDCAW) versus the simplest alternative of assigning calls on a first come first served (FCFS) basis to the first available CSR that is trained to handle the call. Other assignment rules could certainly be considered as well, but this comparison is intended to provide some insights into if an optimization approach should be considered and which formulation performs the best when it should be considered. In addition to the performance measures used by the optimization problems themselves, several other measures that are important in call center operations are evaluated. In particular, as noted earlier, customer service level is perhaps the most common measure used in call centers and should therefore be considered in any such study. Koole and Mandelbaum (2002) state that the industry standard for telephone services seems to be the 80/20 rule, under which at least 80% of the customers must wait no more than 20 seconds. It was noted earlier that this could be modeled as the constraint described in equation (3.14) above.

However, in this experiment we found that this constraint leads to many of the optimization problems being infeasible. This may be partially due to the relatively small size of the problem (that is, only 3 CSRs), but this requires further investigation. To address this, we drop the constraint (3.14) of maintaining a service level of 80/20 and simply evaluate and report the service level obtained by solving the optimization problems.

The variable parameters in the experiment are the scheduling procedure, utilization rate, and service time distribution. The scheduling procedure is one of TFT, MFT, MDWAC, or FCFS. The agent utilization is either 85%, 90%, or 95%, and there are three cases for serving time distribution that reflect different types of specialization for CSRs. There are therefore a total of 36 possible settings. However, with Case I service time distribution where each CSR is able to only work with one type of call, the MDWAC procedure may be omitted because any calls' assigning solutions will result in the same objective function value. This call center is therefore studied under 33 different scenarios. For 24 of those scenarios an IP must be solved each time a new call arrives or a call is completed by a CSR. All of these optimization problems are solved using a LINGO 9.0 commercial solver.

3.2.2 Discussion of Results

As noted above, the call center's performance is observed for a total of 33 scenarios. The observed performance measures include the maximum number of calls (MaxN), the average number of calls in system (AvgN), the average flow time of a call (AvgF), the maximum flow time of a call (MaxF), the average waiting time for a call (AvgW), the maximum waiting time for a call (MaxW), the maximum deviation of assigned workload (Maxcldiv), and the call center service level (SL). The observed data are compared and analyzed to find the effect of using the various scheduling methods on the performances of the inbound call center under different call center characteristics (that is, different agent utilization rate and different serving time scheme as described above).

The objective of these experiments is to determine under what conditions the assignment of incoming calls is sufficiently complex to warrant an optimization approach, and in those scenarios determine which of the three objective functions considered performs the best.

In the case of each CSR being skilled to work with only one type of call there is no difference in the call center's performances among the use of different assignment rules regardless of the level of agent utilization rate. This is a trivial observation since in this extreme case there is no flexibility for assigning calls so simply using a FCFS rule to assign calls is as good as any optimization approach. Thus, in the event that a call center is set up in this fashion there is no call for an optimization approach.

The second extreme case is more interest but still relatively simple. Here all the CSRs have identical serving time and are skilled to work with any types of call. We might again expect a simple FCFS rule to perform well because with identical CSRs there is never a benefit to holding a call for a CSR that is currently busy. However, our numerical results indicate that for a very important special case optimization does provide significant benefit. In particular, the numerical results reported in Table 3.3 indicate that if the goal is to maximize the service level then it is beneficial to assign calls by solving an optimization problem. This table shows the actual service level obtained by using FCFS or solving the TFT or MDCAW optimization problems, as well as the percentage improvement of the optimization approach over FCFS. Similar observation can be made about minimizing the average number of calls in system, where optimization with a flow time objective performed significantly better than FCFS for both low and intermediate utilization. We finally note from Table 3.4 that if the utilization is high and the goal is to either optimize the load balance between CSRs or the service level, then solving an optimization problem MDCAW has the best performance. Note that the table reports the service level (SL), the maximum deviation of assigned workload (Maxcldiv) and the percentage improvement of each optimization problem solution over FCFS.

Table 3.3: Comparison of Service Level using Optimization (TFT or MDCAW) versus FCFS

CSR		*TFT*		*MDCAW*	
Utilization	*FCFS SL*	*SL*	*Change*	*SL*	*Change*
85%	31.3	40.6	30.0%	34.4	10.0%
90%	9.1	15.2	66.7%	15.2	66.7%
95%	10.3	15.4	49.9%	18.0	75.0%

Table 3.4: Comparison of Assignment Policies for Case II Skill Distribution and High Utilization

	FCFS	*MDCAW*		*TFT*		*MFT*	
SL	10.3	18.0	74.8%	15.4	49.5%	12.8	24.3%
Maxcldiv	113.3	33.3	70.6%	77.3	31.8%	156.3	-38.0%

We now turn our attention to the third case of CSR skill level distribution, where each CSR can work on any type of call, but at different service rate. This is the most complex and realistic scenario, and optimization can be expected to be more beneficial in this case. Indeed, our data indicates that FCFS is competitive only when the agent utilization is low.

An overview comparison of different scheduling methods is given in Table 3.5, where the performance as measured by each of the seven measures is identified as similar if they percentage difference is less than 3% (denoted with symbol ~), preferred if the percentage improvement is greater than 3% but less than 5% (denoted with symbol $\succ$), and much preferred if the improvement is greater than 5% (denoted with symbol $\succ\succ$). The last column also identifies if an optimization approach is useful, and in those cases which type of objective function performs the best.

For low utilization, the optimization does not yield significant benefits for any flow time related performance measure, number of calls in system, or service level objective functions. The only exception is when the manager wants to optimize the load balance between CSR. At any utilization level, such load balance is best achieved through solving problem MDCAW (see Table 3.6). Note that this is different than the case of identical CSRs, where solving MDCAW is only useful at high utilization.

Table 3.5: Comparison of Call Scheduling Methods for Case III Skill Distribution.

Performance	*Utilization*	*Preferences Among Scheduling Methods*	**Optimization?**
Min *AvgN*	85 %	FCFS $\succ\succ$ MDCAW $\succ\succ$ TFT $\succ\succ$ MFT	No
	90 %	TFT $\succ\succ$ MDCAW ~ FCFS ~ MFT	Flow objective
	95 %	TFT $\succ\succ$ MDCAW $\succ\succ$ MFT $\succ\succ$ FCFS	Flow objective
Min *MaxF*	85 %	MDCAW $\succ\succ$ FCFS $\succ\succ$ MFT $\succ$ TFT	Balance objective
	90 %	MFT $\succ\succ$ MDCAW ~ TFT $\succ\succ$ FCFS	Flow objective
	95 %	TFT $\succ$ FCFS ~ MFT $\succ$ MDCAW	Flow objective
Min *AvgF*	85 %	FCFS $\succ\succ$ TFT $\succ\succ$ MDCAW $\succ\succ$ MFT	No
	90 %	TFT $\succ\succ$ MFT $\succ$ MDCAW $\succ$ FCFS	Flow objective
	95 %	TFT $\succ\succ$ MFT $\succ\succ$ MDCAW $\succ\succ$ FCFS	Flow objective
Min *MaxW*	85 %	FCFS $\succ\succ$ MDCAW $\succ\succ$ MFT $\succ\succ$ TFT	No
	90 %	FCFS $\succ\succ$ TFT ~ MFT $\succ\succ$ MDCAW	No
	95 %	FCFS $\succ\succ$ TFT $\succ\succ$ MFT $\succ\succ$ MDCAW	No
Min *AvgW*	85 %	TFT $\succ\succ$ MDCAW $\succ\succ$ MFT $\succ\succ$ FCFS	Flow objective
	90 %	TFT ~ MFT $\succ$ MDCAW $\succ\succ$ FCFS	Flow objective
	95 %	TFT $\succ\succ$ MFT $\succ\succ$ MDCAW $\succ\succ$ FCFS	Flow objective
Min *Maxcldiv*	85 %	MDCAW $\succ\succ$ FCFS $\succ\succ$ TFT $\succ\succ$ MFT	Balance objective
	90 %	MDCAW $\succ\succ$ FCFS $\succ\succ$ MFT $\succ\succ$ TFT	Balance objective
	95 %	MDCAW $\succ\succ$ MFT $\succ\succ$ FCFS $\succ\succ$ TFT	Balance objective
Max *SL*	85 %	FCFS $\succ\succ$ MDCAW $\succ\succ$ TFT $\succ\succ$ MFT	No
	90 %	MFT ~ TFT ~ MDCAW $\succ\succ$ FCFS	Flow objective
	95 %	MFT $\succ\succ$ TFT $\succ\succ$ MDCAW $\succ\succ$ FCFS	Flow objective

Table 3.6: Comparison of CSR Work Balance for MDCAW versus FCFS

CSR Utilization	*FCFS Maxcldiv*	*MDCAW*	
		Maxcldiv	*Change*
85%	169.3	60.3	64.4%
90%	103.3	88.3	14.5%
95%	153.3	31.7	79.3%

For other situations, that is, the utilization is intermediate or high and we are not primarily concerned with CSR load balance, solving the optimization problems with flow time objectives results in the best performance (See Table 3.7). Solving problem TFT usually results in better performance than MFT, that is, minimizing the total flow time at each point in time is preferable to minimizing the maximum flow time at each point in time. The only exceptions to this are when the agent utilization rate is intermediate and the objective function is to minimize the maximum flow time and the agent utilization rate is intermediate or high and the objective function is to maximize service level. The maximum flow time objective results in the best performance for those two scenarios (see Table 3.8).

Table 3.7: Comparison of Assignment Policies for Case III Skill Level Distribution

CSR Utilization	*Performance measure*	*FCFS*	*TFT*		*MFT*		*MDCAW*	
90%	SL	18.2	27.3	50.0%	27.3	50.0%	27.3	50.0%
	MaxN	6.0	4.0	33.3%	5.0	16.7%	5.0	16.7%
	AvgF	235.1	182.8	22.2%	216.1	8.1%	223.7	4.8%
	MaxF	578.0	466.0	19.4%	419.0	27.5%	464.0	19.7%
	AvgW	91.1	61.9	32.0%	61.9	32.0%	66.9	26.6%
95%	SL	12.8	25.6	100.0%	30.8	140.0%	17.9	40.0%
	MaxN	7.0	5.0	28.6%	6.0	14.3%	7.0	0.0%
	AvgF	356.7	269.9	24.3%	296.5	16.9%	337.9	5.3%
	MaxF	737.0	701.0	4.9%	759.0	-3.0%	788.0	-6.9%
	AvgW	188.4	129.0	31.6%	154.0	18.3%	164.9	12.5%

Table 3.8: Comparison of TFT versus MFT for Case III Skill Level Distribution

	CSR Utilization					
	85%		*90%*		*95%*	
Performance measure	*TFT*	*MFT*	*TFT*	*MFT*	*TFT*	*MFT*
SL	43.8	34.4	27.3	27.3	25.6	30.8
MaxN	4.0	4.0	4.0	5.0	5.0	6.0
AvgN	0.6	0.8	0.9	1.0	2.0	2.5
MaxF	578.0	561.0	466.0	419.0	701.0	759.0
SumF	6207.0	7114.0	6034.0	7130.0	10527.0	11563.0
AvgF	194.0	222.3	182.8	216.1	269.9	296.5
AvgW	50.1	75.2	61.9	61.9	129.0	154.0

We conclude this section with some general managerial insights that are indicated by our numerical results. First, optimization appears to be most valuable when CSRs are specialized but also trained for calls that are not within their specialization. This is likely to be the situation for most call centers. Second, optimization appears to be more valuable at higher utilization rates, such as the 90%-95% rates that are commonly observed for call centers. Third, solving the MDCAW problem performs well under almost any scenario when the objective is to balance load between CSRs and it generally provides a good service level. Fourth, minimizing the total flow time is most often preferable to minimizing the maximum flow time. Indeed, minimizing the maximum flow time locally, that is, each time a call arrives or departs, often leads to higher overall maximum flow time than minimizing the total flow time locally. Thus, the optimization approach proposed here can improve call center performance under realistic scenarios, and both the TFT and MDCAW problems are useful, depending on what performance measures are most important to call center management.

3.3 Model Conclusion

Call centers are employed by numerous businesses to conduct many of their customer interactions. A key operation in such call centers is scheduling of inbound calls, which involves both assigning calls to CSRs and sequencing the calls waiting for each CSR.

In this chapter, we show that the performance of certain call centers, as measured by service levels, call flow times, and CSR load balance, can be significantly improved by employing an optimization approach for such inbound call scheduling.

The findings reported indicate that optimization is most valuable under realistic scenarios involving specialized but broadly trained CSRs and high call center utilization rates. In particular, solving the MDCAW problem performs well under almost any scenario when the objective is to balance load between CSRs and it generally provides a good service level. Minimizing the total flow time also results in good performance on most performance measures, including flow time related measures and service level; and is the best scheduling method with respect to most performance measures when the utilization is high and CSRs are specialized but broadly trained.

However, the solution approach, we have employed, that is solving each optimization problem independently using a commercial solver, is not fast enough to be used in a real-time ACD system. In the next chapter we will focus on how to improve the solution algorithms. Several directions will be considered for obtaining an optimal or near-optimal schedule under the tight time constraints that are inevitable when using this approach in real time.

CHAPTER 4. SOLUTION TECHNIQUES

In Chapter 3 our findings have shown that the performance of the call centers can be significantly improved by employing an optimization approach to schedule inbound calls. The solution approach used tends to be impracticable under a tight time constraint, as is the case in a real time ACD system. Various approaches can be taken to overcome this problem; however, there is no one size fits all solution technique that is effective for every problem. We need to design a combination of techniques that are suitable for the inbound call scheduling problem. Our aim is "effective" solution techniques which are suitable for inbound call scheduling problems. Frequently, when deciding on a solution technique, we experience a tradeoff between quality of the solutions and its speed. In this research study, we describe an "effective" solution technique as a technique that allows our IPs to generate the same or better quality solution than that obtained from FCFS within a reasonable solution time that allows for real time scheduling of calls. This makes the optimization a competitive approach for scheduling inbound calls.

In Chapter 2, several ideas such as reformulation and combining general LP based BB method with some bound tightening methods to speed up the solution process have already been introduced. In this research, four promising IP solution techniques are investigated: 1) IP Reformulation, 2) Lagrangian Relaxation and Duality, 3) Cutting Plane Algorithm, and finally 4) Solving IP with Initial Solution (herein referred to as Re-optimization Approach)

4.1 Proposed Solution Techniques

4.1.1 IP Reformulation

One of the most common causes of a large runtime is the size of the formulation. As an effort to reduce number of decision variables, the original IP models were reformulated by using different decision variables. By breaking our call scheduling problem into two separate simpler structured components, that is, the assignment problem and the sequencing problem and introducing new decision variables, number of decision variables is changed from m^2n to $m(m+n)$, where m and n are the number of calls and number of CSRs, respectively. For the reformulated IP, we introduce new decision variables as follows:

$$U_{ij}(t) = \begin{cases} 1 & \text{if call } i \text{ is assigned to CSR } j, i \in I, j \in J, \\ 0 & \text{otherwise.} \end{cases}$$

$$V_{iq}(t) = \begin{cases} 1 & \text{if call } i \text{ is sequenced to be served by CSR at position } q, i \in I, q \in Q, \\ 0 & \text{otherwise.} \end{cases}$$

It is clear that for any values of *m* and *n,* except when *m* or *n* is equal to 1, which is very unlikely to be the case of any call centers, the number of decision variables of the new formulation will be less than or equal to the original IPs. The difference in the size of the formulation increases exponentially, when the call center becomes bigger. This makes the reformulated IP a viable alternative.

Now given the queue position, assignment, and the processing time, the following supporting variables and the CSR service time condition can be derived and will be utilized in the new optimization models.

$$Q_i(t,V) = \sum_{q=1}^{m(t)} qV_{iq}(t), \tag{4.1}$$

$$J_i(t,U) = \sum_{j=1}^{n(t)} jU_{ij}(t), \tag{4.2}$$

$$QL_i(t,U,V) = \sum_{j=1}^{n(t)} \sum_{k=1}^{m(t)} P_{kj} U_{kj}(t) U_{ij}(t) \chi_{\{Q_k(t,V) \le Q_i(t,V)\}}, \tag{4.3}$$

$$SA_i(t,U,V) = w_i(t) + \sum_{j=1}^{n(t)} [r_j(t) U_{ij}(t)] + QL_i(t,U,V), \tag{4.4}$$

$$Cl_j(t,U) = ACl_j(t_0) + \sum_{i=1}^{m(t)} P_{ij} U_{ij}(t), \tag{4.5}$$

$$ACl_j(t,U,V) = ACl_j(t_0) + \sum_{k=1}^{m(t)} P_{kj} U_{kj}(t) \chi_{\{Q_k(t,V) \le S_j(t,U,V)\}}, \tag{4.6}$$

$$s_j(t,U,V) = \max\left\{ s_j(t,U,V) : r_j(t) + \sum_{k=1}^{m(t)} P_{kj} U_{kj}(t) \chi_{\{Q_k(t,V) \le S_j(t,U,V)-1\}} \le t_1 - t, s_j(t,U,V) \in Z^+ \right\} \tag{4.7}$$

Where $\chi_{\{f(t) \le g(t)\}} = \begin{cases} 1 & \text{if } f(t) \le g(t) \\ 0 & \text{otherwise.} \end{cases}$

With the above notation, we can now reformulate the mathematical programming problems for scheduling inbound calls in call centers as follow:

Reformulated Problem TFT (RTFT)

$$\min_{\mathrm{U,V}} \sum_{i=1}^{m} F_i(t,U,V) = \sum_{i=1}^{m}\left(SA_i(t,U,V) + \sum_{j=1}^{n(t)} P_{ij}U_{ij}(t)\right), \tag{4.8}$$

Subject to

$$\sum_{j=1}^{n(t)}\sum_{q=1}^{m(t)} V_{iq}(t)U_{ij}(t) = 1, \forall i, \tag{4.9}$$

$$\sum_{i=1}^{m(t)} V_{iq}(t)U_{ij}(t) \le 1, \forall j, q\ . \tag{4.10}$$

Constraint (4.9) ensures that each call is served exactly once by exactly one CSR. Constraint (4.10) ensures that not more than one call can be served at a time by a CSR.

Reformulated Problem MFT (RMFT)

$$\min_{\mathrm{U,V}} \max_{i} \left\{ F_i(t,U,V) \right\} = \max_{i}\left\{ \sum_{i=1}^{m}\left(SA_i(t,U,V) + \sum_{j=1}^{n(t)} P_{ij}U_{ij}(t)\right)\right\}. \tag{4.11}$$

This is subject to constraints (4.9) to (4.10).

Reformulated Problem MDCAW (RMDCAW)

$$\min_{\mathrm{U,V}} \max_{j} \left| Cl_j(t,U) - \frac{1}{n(t)} \sum_{j=1}^{n(t)} Cl_j(t,U) \right|, \tag{4.12}$$

This is subject to constraints (4.9) to (4.10).

It is wise to note that introducing new decision variables does not affect the mathematical equation of the service level constraint and of the load balance constraint, thus both constraints remain unchanged. The above formulations can be improved by using separate constraints to tighten the feasible region, that is, we replace constraint (4.9) with the stronger valid equalities (4.13) and (4.14).

$$\sum_{q=1}^{m(t)} V_{iq}(t) = 1, \forall i, \qquad (4.13)$$

$$\sum_{j=1}^{n(t)} U_{ij}(t) = 1, \forall i, \qquad (4.14)$$

4.1.2 Lagrangian Relaxation and Duality

Lagrangian relaxation and duality is a well-accepted optimization technique that researchers use to obtain boundaries for optimization problems. There is motivation for applying stronger lower bound values than those achieved by the continuous relaxation (CR) approach. Adding of the lower bound to the formulated IP will tighten the problem and results in a smaller polytope, which in turn often helps speed up the solution process. We expect not only the decreasing the solution time, but also improved solution quality.

Considering problem TFT

$$\min_{\mathbf{X}} \sum_{i=1}^{m} F_i(t, X),$$

Subject to

$$\sum_{q=1}^{m(t)} \sum_{j=1}^{n(t)} X_{ijq}(t) = 1, \forall i,$$

$$\sum_{i=1}^{m(t)} X_{ijq}(t) \le 1, \forall j, q.$$

Among the two type of constraints in the above IP formulation, that are assignment constraints and sequencing constraints, we decide to drop the sequencing constraints $\sum_{i=1}^{m(t)} X_{ijq}(t) \le 1, \forall j, q$ and include them in the objective function with the Lagrangian multipliers. The relaxation obtained is regarded as an assignment problem which is a relatively easy problem to solve. Then, we derive the Lagrangian dual.

Lagrangian Dual of Problem TFT (LDTFT)

$$\max_{\lambda}\left[\min_{\mathbf{X}}\sum_{i=1}^{m}F_i(t,X)+\sum_{q=1}^{m(t)}\sum_{j=1}^{n(t)}\lambda_{jq}(\sum_{i=1}^{m(t)}X_{ijq}(t)-1)\right], \qquad (4.15)$$

Subject to

$$\sum_{q=1}^{m(t)}\sum_{j=1}^{n(t)}X_{ijq}(t)=1, \forall i, \qquad (4.16)$$

Where λ is the vector of Lagrangian multipliers.

With the same procedure, we can derive

Lagrangian Dual of Problem MFT (LDMFT)

$$\max_{\lambda}\left[\min_{\mathbf{X}}\max_{i}\{F_i(t,X)\}+\sum_{q=1}^{m(t)}\sum_{j=1}^{n(t)}\lambda_{jq}(\sum_{i=1}^{m(t)}X_{ijq}(t)-1)\right], \qquad (4.17)$$

Subject to constraints (4.16).

Lagrangian Dual of Problem MDCAW (LDMDCAW)

$$\max_{\lambda}\left[\min_{\mathbf{X}}\max_{j}\left|Cl_j(t,X)-\frac{1}{n(t)}\sum_{j=1}^{n(t)}Cl_j(t,X)\right|+\sum_{q=1}^{m(t)}\sum_{j=1}^{n(t)}\lambda_{jq}(\sum_{i=1}^{m(t)}X_{ijq}(t)-1)\right], \qquad (4.19)$$

Subject to constraints (4.16).

Due to their simplicity, sub-gradient algorithms are often used as a tool to obtain solutions of Lagrangian dual problems. In this research study, we will use the sub-gradient algorithm method to solve LDTFT, LDMFT and LDMDCAW problem. Note that to make thing simple, the following specifications are made in our research study:

1) Initial Lagrangian Multiplier U_0 is set to 0.

2) A simple square summation but not summable step rule $\alpha_k = 1/k$ is used, where k is the iteration number. Although the convergence is slow, the algorithm is guaranteed to converge to the optimal Solution.

3) Several criteria can be used to terminate sub-gradient algorithm iterations. For example, the iteration should be terminated if the optimal solution is found or if the objective value is not improved within a number of iterations, or if the maximum solution time is reached. For our inbound call scheduling problem, setting the maximum solution time would be a practical and reasonable stopping point when the optimal solution for the optimization problem is not found. Therefore the maximum solution time for termination sub-gradient algorithm is when there is new call coming in to system and triggering the new optimization.

4.1.3 Cutting Plane Algorithm

The idea of adding valid inequalities as cut constraints to IP has been used to solve a great number of problems that have vast solution spaces. Although adding more constraints might increase the complexity to the formulation, it can significantly reduce the size of solution space.

As previously mentioned, our call scheduling problem is the mixture of sequencing problems and assignment problems. Any valid inequalities of assignment problem and sequencing problem will therefore also be valid for the calls scheduling problem.

In Farias and Nemhauser (2001), a family of valid inequalities for the generalized assignment polytope is introduced. Specifically, they introduce the following multiple-choice GAP (MGAP) constraint:

"at most one of $x_{i1,\dots,}\, x_{in,}$ is positive for $\forall i \in M$."

As mentioned by Farias and Nemhauser (2001), although the MGAP constraints in general are not facet-defining, they define facets of a polytope of a continuous relaxation (CR) of generalized assignment problem. The demonstration through computational experimentation revealed that MGAP constraints are effective cuts for solving generalized assignment problem when incorporated into a branch and cut scheme.

By slightly modifying the MGAP constraint to match with our models, we obtain at most one of X_{ijq} is positive for $\forall i \in M$.

$$\sum_{q=1}^{m(t)}\sum_{j=1}^{n(t)} X_{ijq}(t) \le \underset{X}{Max}(X_{ijq}(t)), \forall i \tag{4.19}$$

If we append the above valid inequalities as cut constraints (4.19) for problem TFT, problem MFT, and problem MDCAW, we obtain problem **CTFT, CMFT,** and **CMDCAW**, respectively. With the new formulations, we expect the solution time to be decreased since the cuts together with the sequencing constraints will cut off infeasible fractional solutions when performing branch and cut.

4.1.4 Re-optimization

As previously mentioned, to scheduling inbound calls in the call centers with the proposed optimization approach, an IP must be solved each time a call arrives or is completed by a CSR. Therefore the re-optimization process will be repeatedly performed during the day.

To make use of the available information, that is the previous optimal scheduling solution, both in terms of quality of the solution and the computational efficiency, is to use the previous optimal solution as a basis and re-optimize for new scheduling solutions.

Our idea is that by fixing the new schedule of all the calls that have been in the call center and have not been served by CSRs with their associated previous schedule from optimization and using FCFS approach to schedule the new calls (if there is any). This way, we can obtain the fast and feasible scheduling solutions which later will be referred to as the heuristic feasible scheduling solutions. We then compare the obtained solutions with FCFS. The better scheduling solutions are used as the initial feasible scheduling solutions of the branch and bound procedure to produce the new optimal schedule. Inputting a good initial feasible solution to the problem is a boost to computation since it increases pruning potential. Although the optimal scheduling is not guaranteed, incorporating heuristic procedures into optimization algorithms would still be an effective approach to obtain the fast and reasonable scheduling solutions.

The proposed composite procedure can be outlined as follows:

Initialization: Given the scheduling solutions from the previous optimization ($\overline{X}^{1}$).

Step 1: Obtain the heuristic feasible scheduling solutions ($\overline{X}^{1'}$) by fixing the initial values of the calls that have been in the call center and have not been served by CSRs to their associate scheduling solutions from $\overline{X}^{1}$. For new calls (if there are any) using FCFS approach to obtain their initial values. This step is referred as the HEU method.

Step 2: Compare call center performances offered by $\overline{X}^{1'}$ with those offered by FCFS scheduling solutions ($\overline{X}^{FCFS}$), choose the scheduling solutions which can provide the better performance as the initial feasible scheduling solutions ($\overline{X}^{1*}$).

Step 3: With $\overline{X}^{1*}$, use optimization to obtain new scheduling solutions ($\overline{X}^{2}$) using the formulated IPs.

If the IP employed in step 3 is TFT, in this research, we will refer to the new method as ROTFT. In the same fashion, we derive ROMFT, and ROMDCAW.

The proposed re-optimization approach is very simple, practical to implement, and guarantee to be as good or better solutions than those from FCFS. To apply these approaches to schedule inbound calls under tight time constraint, we might add a condition such as if the optimal solutions could not be found within *t* seconds or *i* iterations, use the best feasible solution found so far. Although $\overline{X}^{1*}$ is a very fast and feasible schedule to obtain, to schedule calls with $\overline{X}^{1*}$, the examination of the solution quality is necessary. The above procedure is depicted in figure 4.1

4.2 Numerical Example

In the preceding section, we proposed 4 solution techniques for our inbound calls scheduling problem. In order to investigate the efficacy of the proposed solution techniques, two phases of the experiments were designed and constructed. The first phase is the preliminary experiment which aims to determine the potential of the solution techniques. The preliminary experiment and its result are described in more detail in appendix B. The preliminary result indicated that Lagrangian Relaxation and Duality, Cutting Plane Algorithm, and Re-optimization approach are good approaches to handling the inbound call scheduling problem.

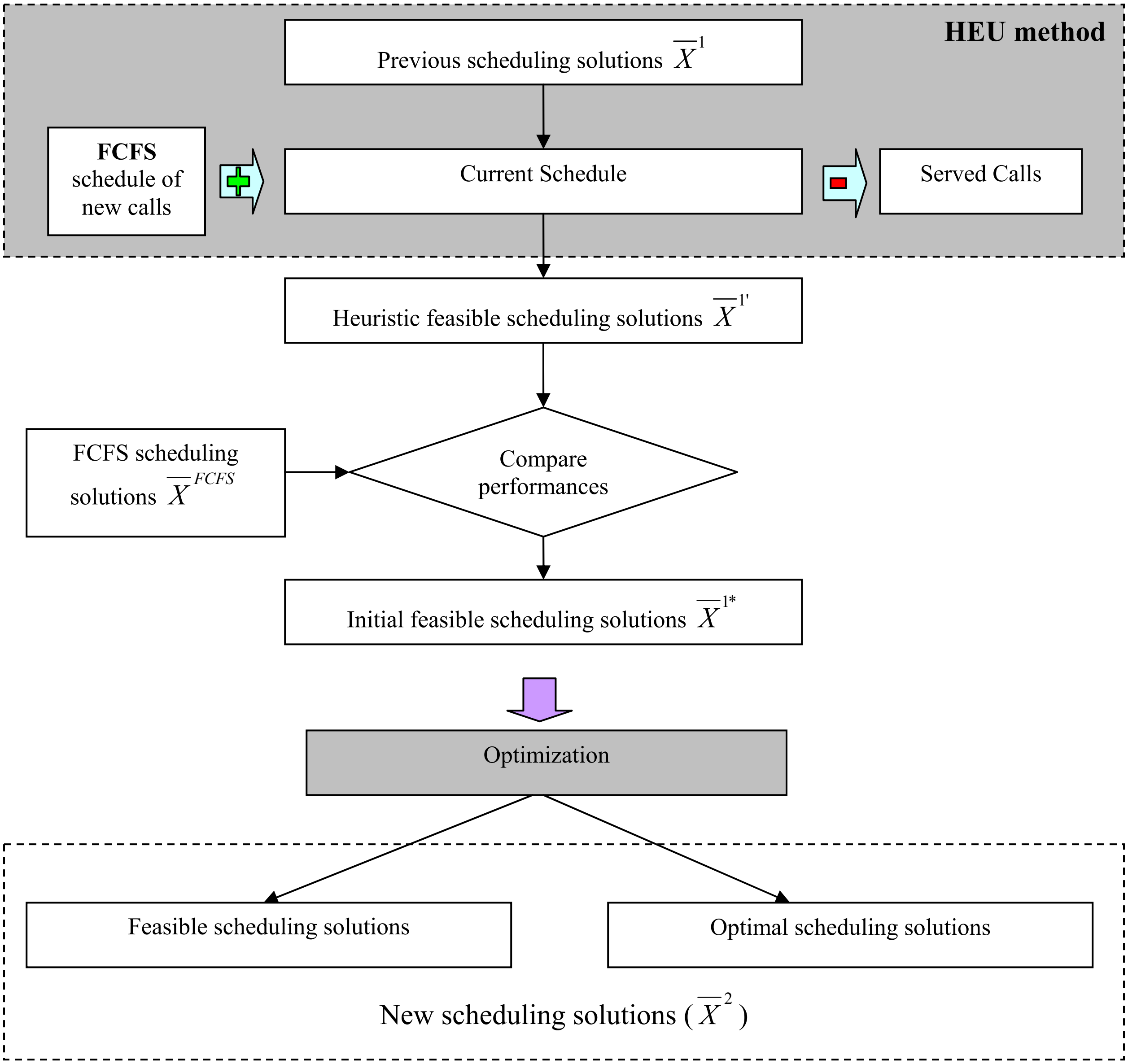
HEU method
Previous scheduling solutions $\overline{X}^{1}$
FCFS schedule of new calls
Current Schedule
Served Calls
Heuristic feasible scheduling solutions $\overline{X}^{1'}$
FCFS scheduling solutions $\overline{X}^{FCFS}$
Compare performances
Initial feasible scheduling solutions $\overline{X}^{1*}$
Optimization
Feasible scheduling solutions
Optimal scheduling solutions
New scheduling solutions ($\overline{X}^{2}$)

Figure 4-1. Re-Optimization Approach

The second phase (the actual experiment) was set up to examine the effect of using the various proposed solution techniques. As recommended from the preliminary result, we chose to concentrate our study on Lagrangian Relaxation and Duality, Cutting Plane Algorithm, and Re-optimization approach. In addition to these promising solution techniques, we are also interested in exploring the quality of heuristic feasible scheduling solutions. Hence, let us include HEU method, the logic of obtaining heuristic feasible scheduling solutions, as another solution technique in our focus.

The objective values of the original IPs consisting of: the total flow time (SumF), the maximum flow time (MaxF), and the maximum deviation of assigned workload (Maxcldiv) along with the associated solution times were recorded, compared and analyzed to identify the effective scheduling methods.

In this section, we explain the experimental design, the data generation, and the discussion of result of the second phase experiment.

4.2.1 Experiment Design

30 different realistic call center's parameter settings were simulated to represent various call centers characteristic, various calls and CSRs attributes at various time periods during the day. Those call center' parameter settings include the time horizon considered t-t_1, number of calls $m(t)$, call type and its waiting time $w_i(t)$, number of CSRs $n(t)$, initial actual cumulative workload $Acl_j(t_0)$ and remaining serving time $r_j(t)$ of each CSR, and the server dependent processing time P_{ij}..

Given the parameter settings described above, the experiment compares the call center's performances and the solution time among the different scheduling methods. The scheduling method is one of the original three mathematical programming models (that is, TFT, MTF, and MDCAW), the logic of obtaining heuristic feasible scheduling solutions (that is, HEU), and the new purposed solution techniques (that is, LDTFT, LDMFT, LDMDCAW, CTFT, CMFT, CMDCAW, ROTFT, ROMFT, and ROMDCAW), and FCFS. There are therefore a total of 390 possible scenarios. All of these optimization problems are solved on an INSPIRON B120 with a LINGO 10.0 commercial solver.

4.2.2 Data Generation

The call centers' setting in this section is similar to the model call center used in section 3.2 with these differences:

1) Utilization rate

We have learnt from our numerical example in chapter 3 that optimization approach appears to be more valuable at higher utilization rates. Therefore instead of varying the call centers' utilization rate of the call center from low (85%), to intermediate (90%), and high (95%), we randomly generate the value of the utilization rates to be in the range of 90% to 100%.

2) Server dependent processing time P_{ij}

We assume CSRs are skilled to work with any types of call but at the different service rate. Using the processing time information from an actual operational call center where the minimum processing time is 4 seconds and the maximum processing time is 879 seconds, we arbitrarily generate the average processing time for each type of call between the minimum and the maximum. We then deviate from the average processing time to get the server dependent processing times for each type of call by randomly generating data from Exponential Distribution.

3) Number of CSR $n(t)$,

Instead of fixing the number of CSRs to 3, for this experiment we randomly generate 30 integer numbers between 4 and 9 to represent the numbers of CSRs in 30 call centers' settings.

With the parameters mentioned above, we calculate the call arrival rates that satisfy the randomly generated utilizations. Like the call center described in section 3.2, we continue to assume that the time between call arrivals is exponentially distributed and there are three types of calls referred as Type I, Type II, and Type III. The call inter-arrival times are generated based on the calculated arrival rates, while the types of calls are randomly selected among the three types of calls.

4) Time horizon considered t- t_1

As explained in section 4.1.4, the Re-optimization approach requires an initial feasible scheduling solution to start an optimization. Therefore we specify a 30-minute pilot run of each experiment setting. At the beginning of the pilot run t_{nitial}, there are calls being serviced by CSRs with the associated remaining serving time $r_j(t_{initial})$. We assume that during this 30-minute period, Re-optimization approach is used to schedule calls in the system. The re-optimization of call scheduling is triggered each time there is a new call arrived into the call center system. For each optimization, the scheduling solution, solving time, and number of iterations are solved and recorded. Call center's parameter settings for the subsequent optimization are also derived.

As for each CSR's initial actual cumulative workload $Acl_j(t_{initial})$, we take into account only the call that the CSR is serving at the beginning of the pilot run. Therefore the initial actual cumulative workload is equal to the processing time of the initial serving call. Since the highest number of call represents the busiest time of CSRs, we rationally select the time horizon considered t- t_1 for our second phase experiment to be when the highest number of call $m(t)$ in the system happens. In the case of having more than one maximum, we simply chose the latest time period. This process yields not only the initial feasible scheduling solution but also the required parameters for all 30 different realistic call center settings.

Two simple data generation functions: 1) RANDBETWEEN ( ) in Microsoft Excel and 2) rexp ( ) in R were used to generate data sets. Table 4.1 summarizes the detail of how the data sets are generated.

Table 4.1: Data Generating Function

Parameter	Data Generating Function	Unit
Utilization rate	RANDBETWEEN (90,100)	%
Average processing time	RANDBETWEEN (4,879)	Seconds
Number of CSRs	RANDBETWEEN (4,9)	CSRs
Inter-arrival time	rexp(*n*,*lamda*)	Seconds
Server dependent processing time	rexp(*n*,1/*mean*)	Seconds
Remaining serving time	RANDBETWEEN (4,9)	Seconds

Note: *n* is the number of observations, *lamda* is the desired call arrival rate, *mean* is the average processing time.

With the above numerical data generation description, the relevant data sets used for all 30 experiment cases are provided in appendix C. Next, we will give the discussion of results.

4.2.3 Discussion of Results

To begin, we briefly review our aim of this research chapter. In this chapter, we intend to identify whether the solution techniques purposed are effective and suitable for inbound calls scheduling problem. An "effective" solution technique, as we define, is a technique that allows our IPs to generate the same or better quality solution than that obtained from FCFS within a reasonable solution time that allows for real time scheduling of calls. Therefore, two main aspects we verified for each solution technique are: 1) whether the quality solution obtained is better than that obtained from FCFS; and 2) whether the good feasible solution can be found within a reasonable solution time. Besides the verification of effectiveness of individual solution technique compared to FCFS, comparisons of quality solutions and solution times among different solution techniques are also included.

Regarding solution time, we recorded the time when the optimal scheduling solutions are found. In the case that optimal scheduling solutions are not found, we recorded the time horizon considered for particular optimization round. However for Lagrangian Relaxation and Duality, the approach of collecting solution time data is slightly different: two solution times are recorded. The first solution time is the total time until sub-optimal scheduling solutions are obtained. The second solution time is the best iteration time of obtaining sub-optimal scheduling solutions.

In determining whether solution techniques perform significantly different than each other, on average, we conducted the one-tailed paired t-test along with the descriptive statistics. The Central Limit Theorem (CLT) permits use of t-test when the population is abnormal. Typically sample size of 30 is used as a rough guideline to claim that the CLT makes the t-test viable.

From the experiment, we found that Lagrangian Relaxation and Duality sometimes did not offer feasible scheduling solutions. In this case, the number of data points obtained would not be enough to apply CLT; hence the normality of data is needed to be assumed. We then provide the frequencies of occurrences as a supplement to capture how well under tight time the solution techniques can provide good feasible solutions.

With the above clarification and the statistical data analysis methods mentioned; findings are presented in 2 main sections: 1) descriptive statistics; and 2) one-tailed paired t-test statistics.

Descriptive Statistics

Tables 4.2, table 4.3 and table 4.4 summarize descriptive statistics of our findings when the objective is to minimize the total flow time, minimize the maximum of flow time, and minimize the maximum deviation of cumulative assigned workload, respectively.

Table 4.2: TFT – Descriptive Statistics

Unit: seconds	FCFS	HEU	TFT	CTFT	LDTFT best time	LDTFT total time	ROTFT
Average SumF	3737.5	2187.8	1793.5	1725.8	1623.2	1623.2	1563.6
Average MaxF	2219.3	1170.2	881.6	819.1	905.5	905.5	722.3
Average MaxclDiv	1980.0	1493.4	1246.2	1335.5	1384.8	1384.8	1357.4
Average solution time	0.0	0.0	33.9	2.2	33.6	73.6	33.0
Average % change of SumF with respect to FCFS	Base value	25%	27%	39%	34%	34%	47%
Frequencies of the occurrence: times (%)							
Feasible solutions found	30 (100%)	30 (100%)	30 (100%)	30 (100%)	23 (77%)	23 (77%)	30 (100%)
No feasible solution found	0 (0%)	0 (0%)	0 (0%)	0 (0%)	7 (23%)	7 (23%)	0 (0%)
Generate better objective value than FCFS	0 (0%)	24 (80%)	25 (83%)	25 (83%)	20 (87%)	20 (87%)	27 (90%)
Generate worst objective value than FCFS	0 (0%)	4 (13%)	4 (13%)	4 (13%)	2 (9%)	2 (9%)	0 (0%)
Generate better objective value than FCFS with same solution time	0 (0%)	24 (80%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)
Generate the best objective value	3 (10%)	6 (20%)	8 (27%)	12 (40%)	11 (48%)	11 (485)	26 (87%)

Table 4.3: MFT – Descriptive Statistics

Unit: seconds	FCFS	HEU	MFT	CMFT	LDMFT best time	LDMFT total time	ROMFT
Average SumF	3549.7	2023.4	1737.4	1695.2	1575.2	1609.3	1569.1
Average MaxF	2061.2	1122.7	893.6	867.6	733.8	733.8	750.0
Average MaxclDiv	1966.6	1456.1	1410.5	1414.2	1438.4	1430.0	1409.3
Average solution time	0.0	0.0	33.9	2.6	29.2	59.0	26.4
Average % change of MaxF with respect to FCFS	Base value	23%	35%	43%	52%	52%	50%
Frequencies of the occurrence: times (%)							
Feasible solutions found	30 (100%)	30 (100%)	30 (100%)	30 (100%)	25 (83%)	25 (83%)	30 (100%)
No feasible solution found	0 (0%)	0 (0%)	0 (0%)	0 (0%)	5 (17%)	5 (17%)	0 (0%)
Generate better objective value than FCFS	0 (0%)	24 (80%)	25 (83%)	24 (80%)	22 (88%)	22 (88%)	26 (87%)
Generate worst objective value than FCFS	0 (0%)	6 (20%)	3 (10%)	4 (13%)	0 (0%)	0 (0%)	0 (0%)
Generate better objective value than FCFS with same solution time	0 (0%)	24 (80%)	0 (0%)	0 (0%)	0 (0%)	22 (88%)	26 (87%)
Generate the best objective value	3 (10%)	12 (40%)	14 (47%)	6 (20%)	17 (68%)	17 (68%)	23 (77%)

Table 4.4: MDCAW – Descriptive Statistics

Unit: seconds	FCFS	HEU	MDCAW	CMDCAW	LDMDCAW	LDMDCAW	RO
					best time	total time	MDCAW
Average SumF	6330.6	5586.9	6876.6	5045.2	2920.8	2958.5	5371.5
Average MaxF	2639.9	2123.6	2257.9	1721.3	1598.1	1614.6	1879.4
Average MaxclDiv	1809.9	1290.2	1053.6	862.0	956.6	956.6	709.4
Average solution time	0.0	0.0	125.9	1.6	56.3	60.9	91.8
Average % change of MaxclDiv with respect to FCFS	Base value	14%	32%	36%	30%	30%	48%
Frequencies of the occurrence: times (%)							
Feasible solutions found	30 (100%)	30 (100%)	29 (97%)	30 (100%)	15 (50%)	15 (50%)	30 (100%)
No feasible solution found	0 (0%)	0 (0%)	1 (3%)	0 (0%)	15 (50%)	15 (50%)	0 (0%)
Generate better objective value than FCFS	0 (0%)	21 (70%)	22 (76%)	24 (80%)	11 (73%)	11 (73%)	27 (90%)
Generate worst objective value than FCFS	0 (0%)	8 (27%)	4 (14%)	4 (13%)	1 (7%)	1 (7%)	0 (0%)
Generate better objective value than FCFS with same solution time	0 (0%)	21 (70%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)
Generate the best objective value	3 (10%)	6 (20%)	13 (45%)	9 (30%)	9 (60%)	9 (60%)	21 (70%)

Summary of findings:

1. Regardless of objective value of interest, all solution techniques provide lower average SumF, lower average MaxF, and lower average MaxclDiv compared to FCFS. More specifically,
   - ROTFT provides the lowest average SumF, followed by LDTFT, CTFT, TFT, HEU, and FCFS.
   - LDMFT provides the lowest average MaxF, followed by ROMFT, CMFT, MFT, HEU, and FCFS.
   - ROMDCAW provides the lowest average MaxclDiv, followed by CMDCAW, LDMDCAW, MDCAW, HEU, and FCFS.

2. Regardless of objective value of interest, HEU can provide scheduling solutions with lower objective values than FCFS without any solution time adding, while for all other techniques more solution time can be expected when solutions with lower objective values are provided.

3. Chance of obtaining better quality scheduling solutions
   - In all experiments, SumF values obtained from ROTFT are better than or same as those obtained from FCFS. Whereas CTFT at 13%, TFT at 13%, and HEU at 13%, and LDTFT at 9% of times are trapped in bad local optimal and result in higher SumF values.
   - In all experiments, MaxF values obtained from ROMFT, and LDMFT are better than or same as those obtained from FCFS. Whereas HEU at 20%, CMFT at 13%, and MFT at 10% of times are trapped in bad local optimal and result in higher MaxF values.
   - In all experiments, MaxclDiv values obtained from ROMDCAW are better than or same as those obtained from FCFS. Whereas LDMDCAW at 7%, CMDCAW at 13%, MDCAW at 14%, and HEU at 27% of times are trapped in bad local optimal and result in higher MaxclDiv values.
   - In 27 of 30 experiments (90%), SumF values obtained from ROTFT are better than those obtained from FCFS. Whereas LDTFT at 87%, CTFT at 83%, TFT at 83%, and HEU at 80% of times can offer better SumF value compared to FCFS.
   - In 22 of 25 experiments (88%), MaxF values obtained from LDMFT are better than those obtained from FCFS. Whereas ROMFT at 87%, MFT at 83%, CMFT at 80%, and HEU at 80% of times can offer better MaxF value compared to FCFS.
   - In 27 of 30 experiments (90%), MaxclDiv values obtained from ROMDCAW are better than those obtained from FCFS. Whereas CMDCAW at 80%, MDCAW at 76%, LDMDCAW at 73%, and HEU at 70% of times can offer better MaxclDiv value compared to FCFS.

4. Chance of obtaining the best objective value
    - In 26 of 30 experiments (87%) the SumF from ROTFT turn out to be the best objective values. Whereas LDTFT at 48%, CTFT at 40%, TFT at 27%, HEU at 20% and FCFS at 10% of times can offer the best SumF value.
    - In 23 of 30 experiments (77%) the MaxF from ROMFT turn out to be the best objective values. Whereas LDMFT at 68%, MFT at 47%, HEU at 40%, CMFT at 20% and FCFS at 10% of times can offer the best MaxF value.
    - In 21 of 30 experiments (70%) the MaxclDiv from ROMDCAW turn out to be the best objective values. Whereas LDMDCAW at 60%, MDCAW at 45%, CMDCAW at 30%, HEU at 20%, and FCFS at 10% of times can offer the best MaxclDiv value.

5. In terms of average solution time,
    - LDTFT total time consumes longest average solution time, followed by TFT, LDTFT best time, ROTFT, and CTFT.
    - LDMFT total time consumes longest average solution time, followed by MFT, LDMFT best time, ROMFT, and CMFT.
    - MDCAW consumes longest average solution time, followed by ROMDCAW, LDMDCAW total time, LDMDCAW best time, and CMDCAW.

6. Chance of finding feasible scheduling solutions,
    - In 7 of 30 experiments (23%) LDTFT is not able to find feasible scheduling solutions. Whereas LDMFT at 17%, LDMDCAW at 50%, and MDCAW at 3% of the times are not able to find feasible scheduling solutions.

To measure the competency among solution techniques, a comparison of descriptive information was also constructed. We, based on our interest, specified 5 key solution techniques performances as shown in table 4.5. The frequencies of occurrences reported in table 4.2, table 4.3, and table 4.4 are compared and analyzed to indentify the best solution techniques under each performance category.

Table 4.5: Descriptive Comparison of Solution Techniques

	Objective value		
	SumF	MaxF	MaxclDiv
Ability to provide the best objective value	ROTFT	ROMFT	ROMDCAW
Ability to provide better objective value compared to FCFS	ROTFT	LDMFT	ROMDCAW
Ability to provide feasible scheduling solutions	FCFS, HEU, TFT, CTFT, ROTFT	FCFS, HEU, MFT, CMFT, ROMFT	FCFS, HEU, CMDCAW, ROMDCAW
Average % change in objective value compare to FCFS	ROTFT	LDMFT	ROMDCAW
Average speed of providing feasible scheduling solutions	FCFS, HEU	FCFS, HEU	FCFS, HEU

Due to the overall performances, we would recommend our Re-optimization approach for scheduling inbound calls in call center as it usually provides good quality solution within tight time regardless of which objective is of interest. Although our evidence shows Lagrangian Relaxation and Duality has higher frequencies of occurrence in finding MaxF value, the concern of its ability to provide feasible scheduling solution and its long solution time makes Re-optimization approach more appealing.

One-tailed Paired T-test Statistics

Tables 4.6, table 4.7 and table 4.8 summarize one-tailed paired t-test result of our findings when the objective is to minimize the total flow time, minimize the maximum of flow time, and minimize the maximum deviation of cumulative assigned workload, respectively. In the tables, we report the statistical comparison of solution techniques whether one solution technique performs better than another at 95% confidence level. The first result in each column is for comparing mean of objective values and the second result is for comparing mean of solution times. P-values of one-tailed paired t-test are provided in Appendix D.

Table 4.6: TFT – Statistical Comparison of Solution Techniques for Mean of Total Flow Time and for Mean of Solution Time at 95% confidence level

Result of Paired T-test	Compare to						
	LDTFT Total time	TFT	LDTFT best time	ROTFT	CTFT	HEU	FCFS
LDTFT total time	= / =	= / >	= / >	> / >	= / >	< / > *	< / > *
TFT	= / < *	= / =	= / =	> / =	= / >	< / > *	< / > *
LDTFT best time	= / <	= / =	= / =	> / =	= / >	< / > *	< / > *
ROTFT	< / < *	< / = *	< / = *	= / =	< / > *	< / > *	< / > *
CTFT	= / < *	= / < *	= / < *	> / <	= / =	< / > *	< / > *
HEU	> / <	> / <	> / <	> / <	> / <	= / =	< / = *
FCFS	> / <	> / <	> / <	> / <	> / <	> / =	= / =

Note that =, <, or > symbols are provided to indicate when one solution technique significantly provided equal, lower, or higher mean value than the other at the 95% confidence level. * indicates if one method is preferable to another.

Table 4.7: MFT – Statistical Comparison of Solution Techniques for Mean of Maximum Flow Time and for Mean of Solution Time at 95% confidence level

Result of Paired T-test	Compare to						
	LDMFT Total time	MFT	LDMFT best time	ROMFT	CMFT	HEU	FCFS
LDMFT total time	= / =	< / > *	= / >	= / >	< / > *	< / > *	< / > *
MFT	> / <	= / =	> / =	> / =	= / >	< / > *	< / > *
LDMFT best time	= / < *	< / = *	= / =	= / =	< / > *	< / > *	< / > *
ROMFT	= / < *	< / = *	= / =	= / =	< / > *	< / > *	< / > *
CMFT	> / <	= / < *	> / <	> / <	= / =	< / > *	< / > *
HEU	> / <	> / <	> / <	> / <	> / <	= / =	< / = *
FCFS	> / <	> / <	> / <	> / <	> / <	> / =	= / =

Note that =, <, or > symbols are provided to indicate when one solution technique significantly provided equal, lower, or higher mean value than the other at the 95% confidence level.

Table 4.8: MDCAW – Statistical Comparison of Solution Techniques for Mean of Maximum Deviation of Cumulative Assigned workload and for Mean of Solution Time at 95% confidence level

Result of Paired T-test	Compare to						
	MDCAW	LDMDCAW total time	LDMDCAW best time	HEU MDCAW	CMDCAW	HEU	FCFS
MDCAW	= / =	= / =	= / =	> / >	= / >	= / >	< / > *
LDMDCAW Total time	= / =	= / =	= / >	= / =	= / >	< / > *	< / > *
LDMDCAW Best time	= / =	= / < *	= / =	= / =	= / >	< / > *	< / > *
ROMDCAW	< / < *	= / =	.= / =	= / =	< / > *	< / > *	< / > *
CMDCAW	= / < *	= / < *	= / < *	> / <	= / =	< / > *	< / > *
HEU	= / < *	> / <	> / <	> / <	> / <	= / =	< / = *
FCFS	> / <	> / <	> / <	> / <	> / <	> / =	= / =

Note that =, <, or > symbols are provided to indicate when one solution technique significantly provided equal, lower, or higher mean value than the other at the 95% confidence level.

Summary of findings:

One-tailed paired T-test shows at 95% confidence level:

1. There is no significant difference among object values obtained from
   - TFT, CTFT, and LDTFT.
   - MFT and CMFT.
   - LDMFT and ROMFT.
   - HEU and MDCAW.
   - MDCAW, CMDCAW and LDMDCAW.
   - LDMDCAW and ROMDCAW.

2. Comparison of Mean of SumF values
   - ROTFT significantly provides the scheduling solutions with lower SumF values than any solution techniques.
   - CTFT significantly provides the scheduling solutions with lower SumF values than HEU and FCFS.
   - HEU significantly provides the scheduling solutions with lower SumF values than FCFS.

3. Comparison of Mean of MaxF values
   - ROMFT and LDMFT significantly provide the scheduling solution with lower MaxF values than any solution techniques.
   - CMFT and MFT significantly provide the scheduling solutions with lower MaxF than HEU and FCFS with 98% confidence.
   - HEU significantly provides the scheduling solutions with lower MaxF than FCFS with 96% confidence.

4. Comparison of Mean of MaxclDiv values
    - ROMDCAW significantly provide the scheduling solution with lower MaxclDiv values than all solution techniques, except LDMDCAW.
    - LDMDCAW and CMDCAW significantly provide the scheduling solutions with lower MaxclDiv than HEU and FCFS.
    - HEU significantly provides the scheduling solutions with lower MaxclDiv than FCFS.

5. There is no significant difference among solution times obtained from
    - TFT, ROTFT, and LDTFT best time.
    - MFT, ROMFT, and LDMFT best time.
    - MDCAW and LDMDCAW total time
    - MDCAW and LDMDCAW best time
    - LDMDCAW best time and ROMDCAW

6. Comparison of Mean of solution times
    - CTFT consumes significantly less solution time than LDTFT, ROTFT and TFT.
    - LDTFT total time consumes significantly more solution time than other solution techniques.
    - CMFT consumes significantly less solution time than LDTFT, ROMFT and MFT.
    - LDMFT total time consumes significantly more solution time than other solution techniques.
    - CMDCAW consumes significantly less solution time than MDCAW, LDMDCAW, and ROMFT.
    - Regardless of objective value of interest, our numerical results show that Lagrangian Relaxation and Duality significantly consumes less solution time if good Lagrangian multipliers can be established.

7. From 1 to 7, we can conclude at 95% confidence level that

When objective is to minimize the total flow time

For SumF value: FCFS>HEU>LDTFT=TFT=CTFT>ROTFT

For solution time: LDTFT total time>TFT=ROTFT= LDTFT best time>CTFT >HEU=FCFS

When objective is to minimize the maximum flow time

For MaxF value: FCFS>HEU>MFT=CMFT>LDMFT=ROMFT

For solution time: LDMFT total time>MFT=LDMFT best time=ROMFT>CMFT >HEU=FCFS

When objective is to minimize the maximum deviation of cumulative assigned workload

For MaxclDiv value:

FCFS>HEU

HEU>CMDCAW>ROMDCAW

HEU=MDCAW

MDCAW=CMDCAW=LDMDCAW

LDMDCAW=ROMDCAW

For solution time:

MDCAW=LDMDCAW total time

MDCAW=LDMDCAW best time

ROMDCAW=LDMDCAW total time

ROMDCAW=LDMDCAW best time

MDCAW>ROMDCAW

LDMDCAW total time>LDMDCAW best time

ROMDCAW>CMDCAW>HEU=FCFS

In addition to the results from one-tailed paired t-test, we indicate if one solution technique is preferable to another by using * symbol. Solution technique A is preferable to Solution technique B when it can either provide better quality scheduling solutions or use less time to provide same or better quality scheduling solutions. Our findings show that Re-optimization approach is the most preferable when objective is to minimize the total flow time. When objective is to minimize the maximum flow time or to minimize the maximum deviation of cumulative assigned workload, Lagrangian Relaxation and Duality could be as preferable as Re-optimization approach if we can improve the ability of finding feasible scheduling solution. However if solution time is a concern to call centers, Cutting Plane Algorithm is the next preferable method as it can significantly provide same solution with less solution time than the original IPs at 95% confidence level.

4.3 Conclusion of Solution Techniques

In this chapter, we show that Re-optimization approach, Lagrangian Relaxation and Duality, and Cutting Plane algorithm are effective solution techniques for inbound call scheduling problems. This finding renders the use of optimization approach for real time scheduling of inbound calls.

The use of valid inequalities as cut constraints allows IPs to quickly find good feasible scheduling solutions. Applying cutting plane algorithm to our inbound calls scheduling problem, the solution time is significantly reduced when the quality of the solutions is unchanged. To obtain better quality solutions, the Lagrangian Relaxation and Duality and the Re-optimization approach might be better option. When solving with subgradient algorithm, the ability to provide feasible scheduling of Lagrangian Relaxation and Duality is in concern. However the ability of finding good scheduling solutions still makes this solution technique meaningful.

It is not surprising that Re-optimization approach shows superior in finding good scheduling solutions. The logic of applying good heuristic feasible scheduling solutions as initial feasible solutions to the IPs not only guarantees that the worst solution than FCFS will never be obtained, it also increases the possibility of escaping from one local optimum. HUE, with no solution time required, has ability to provide better quality solution than FCFS. Hence it is an effective method to find feasible scheduling solutions.

CHAPTER 5. CASE STUDY

While numerical studies in chapter 3 and chapter 4 have demonstrated that the performance of call centers can be improved by employing an optimization approach for scheduling inbound calls, in this chapter, we adopt a case study to investigate the capabilities of solution techniques developed in this research study to handle inbound call scheduling problem of a call center. Based on the call center's characteristic, our research findings in chapter 4, and the limitation of the IP solver software used in this research study, we suggested a composite solution technique for inbound call scheduling and then construct a numerical experiment to compare the call center performances as resulted from using its current inbound call scheduling method with the suggested solution techniques.

5.1 Description of Call Center

The call center in this case study is modeled based on the same actual operational call center used in chapter 3.

As mentioned in chapter 3, within the call center, there are 43 CSRs working under 8 supervisors. Each CSR is scheduled to work 7.5 hours per day excluding break and lunch time. Shifts are planned to meet the incoming call volumes. Only two or three CSRs will share the same schedule in order to spread the available work evenly throughout the center's working hours from 8AM to 6PM. CSRs are skilled to work on multiple types of calls. Most CSRs are specialized in certain types of calls, and hence have faster service rates for those calls, while only minimally trained on others, and hence have slower service rates on those calls. Again although this model call center can handle both inbound and outbound calls only inbound calls are treated in this case study. Note that type of calls is referred as split in this call center.

As described by Barton (2004), when a customer dials a 1-800-###-#### number, the caller is placed into the main voice response unit (VRU) queue. If the caller request to speak with a CSR, then he will go into one of nine Split queues before being transferred. The CSR will then handle the call. The service times were extracted from the CMS system for each Customer Representative for each split. Each CSR is given a skill set (1-4) based on their ability to take that specific type of call. When a call enters a Split queue, the system will attempt to get a CSR according to the following algorithm:

ROUTING LOGIC TO CSR

```
1) Attempt to find all CSRs with a skill set of ‘1’.
IF there is no Level 1 CSR available or CSR is off-shift to assist the customer THEN #2
ELSE CSR will answer the call
2) Attempt to find a CSR that has a ‘2’ rating.
IF there is no Level 2 CSR available or off-shift to assist the customer THEN #3
ELSE CSR will answer the call
3) WAIT 1 second
IF there is no Level 1 or Level 2 CSR available or off-shift to assist the customer THEN
IF time in Split Queue is greater than 45 seconds or Queue contents > 3 THEN #4
ELSE wait 1 second THEN #1
ELSE CSR with Skill Set ‘1’ or ‘2 ‘will answer the call
4) Attempt to find a CSR that has a ‘3’ rating.
IF there is no Level 3 CSR available or off-shift to assist the customer THEN
5) WAIT 1 second
IF there is no Level 1 or Level 2 or Level 3 CSR available or off-shift to assist the customer THEN
IF time in Split Queue is greater than 60 seconds or Queue contents > 5 THEN #6
        ELSE wait 1 second THEN #1
ELSE CSR with Skill Set ‘1’ or ‘2 ‘ or ‘3’ will answer the call
6) Attempt to find a CSR that has a ‘4’ rating.
IF there is no Level 1 or Level 2 or Level 3 or Level 4 CSR available or off-shift to assist the customer THEN #1
ELSE CSR with Skill Set ‘1’ or ‘2 ‘ or ‘3’ will answer the call
```

Since the call center believes if the timely service does not exist, customers will take their business elsewhere, their routing logic depends on the call type and the skill set of the CSR and this logic is referred as ‘WHAT IF’ scenario method.

5.2 Selection of Solution Technique

According to our findings in chapter 4, Re-optimization approach is the most preferable when objective is to minimize the total flow time. When objective is to minimize the maximum flow time or to minimize the maximum deviation of cumulative assigned workload, Lagrangian Relaxation and Duality could be as preferable as Re-optimization approach if we can improve the ability of finding feasible scheduling solution. However if solution time is a concern to call centers, Cutting Plane Algorithm is the next preferable method

Due to the size of the call center used in this case study (i.e., 32 CSRs), we experienced the long solution time when using Re-optimization and the Lagrangian Relaxation and Duality approach. This long solution time issue has become a big concern and prohibited the use of the Re-optimization and the Lagrangian Relaxation and Duality approach. We therefore in this chapter turn our focus to FCFS, HEU, and cutting plane methods since these solution techniques require relatively small solution time.

For the experiment, we proposed a composite procedure as outlined below:

Initialization: Given the previous scheduling solutions ($\overline{X}^{1}$).

Step 1: Obtain the feasible scheduling solutions ($\overline{X}^{1'}$) from the HEU method.

Step 2: Use optimization to obtain new scheduling solutions ($\overline{X}^{CUT}$) by solving IPs with the cutting plane algorithm as solution techniques.

Step 3: Compare call center performances offered by $\overline{X}^{1'}$ with those offered by FCFS scheduling solutions ($\overline{X}^{FCFS}$), and those offered by the optimization ($\overline{X}^{CUT}$), choose the scheduling solutions which can provide the best performance as the scheduling solutions ($\overline{X}^{2}$).

If the IP employed in step 2 is CTFT, in this research, we will refer the new mix approach as MIXTFT. In the same fashion, we derive MIXMFT, and MIXMDCAW. The above mix approach is depicted in figure 5.1

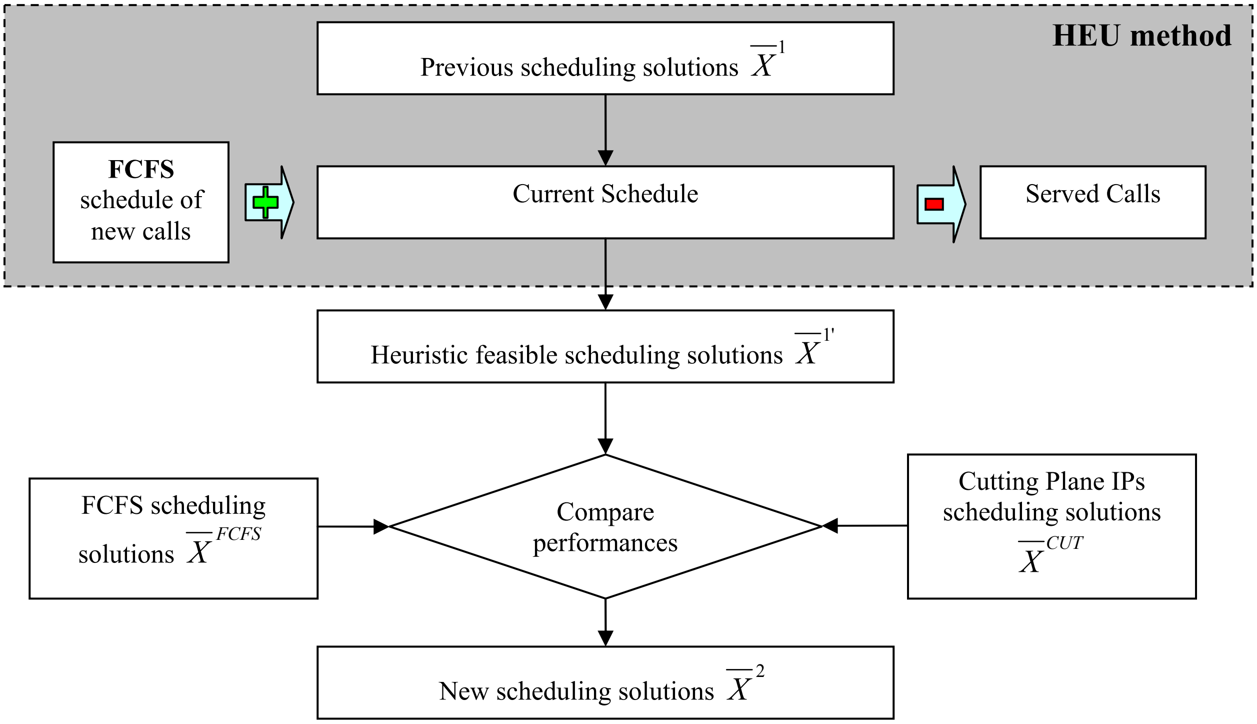
HEU method
Previous scheduling solutions $\overline{X}^{1}$
FCFS schedule of new calls
Current Schedule
Served Calls
Heuristic feasible scheduling solutions $\overline{X}^{1'}$
FCFS scheduling solutions $\overline{X}^{FCFS}$
Compare performances
Cutting Plane IPs scheduling solutions $\overline{X}^{CUT}$
New scheduling solutions $\overline{X}^{2}$

Figure 5-1. Mix Approach

5.3 Numerical Example

In the preceding section, we introduced the mix approaches and also described the step of applying the approaches to solve inbound calls scheduling problem in a call center. In this section, we set up an experiment to compare the performances of the call center resulting from using the mix approaches as the suggested solution techniques and those resulting from using the current routing method as explained in section 5.1.

5.3.1 Experiment Design

As our attempt to construct a modeled call center to be as realistic as possible, most of the parameter settings used in our modeled call center was either directly obtained from or derived from the data extracted by CMS system of the actual system which was reported by Barton (2004). The detail of data generation for these parameter settings is provided in the next section.

5.3.2 Data Generation

This section provides details on how the data were obtained and then manipulated to get the data set in appendix E.

1) Utilization rate

Since the information of agent utilization was not offered by Barton (2004), the utilization rate of intermediate (90%) was randomly selected for this case study.

2) Server dependent processing time P_{ij}

When calls are taken by the CSR, the time is tracked according to what is called After Call Delay Time (or ACD Time). Each CSR handles a call differently, thus ACD times are different for each CSR and for each type of call. In this study we set server dependent processing time (P_{ij}) to the associated average ACD time.

3) Number of CSR $n(t)$

Based on the work shift schedule reported by Barton (2004), there are approximately 32 CSRs available on average. Thus, the number of CSRs $n(t)$ was set to be 32 for this case study.

4) Call Arrival Patterns

As observed by Barton (2004), calls arrive differently depending on the time of day, and the day of week. The number of inbound calls in each split were captured and summarized in daily basis. For this case study, there are nine types of calls, which we simply refer to as Type I, Type II, Type III, Type IV, Type V, Type VI, Type VII, Type VIII, and

Type IX. The percent share for each type of call is approximated and reported with its call arrival rates that satisfy the randomly generated utilizations.

Table 5.1: Percent Share for Each Type of Call

	Utilization Rate
	90%
Type I call's arrival rate: λ_1 (calls/hr)	235
Type II call's arrival rate: λ_2 (calls/hr)	11
Type III call's arrival rate: λ_3 (calls/hr)	8
Type IV call's arrival rate: λ_4 (calls/hr)	59
Type V call's arrival rate: λ_5 (calls/hr)	258
Type VI call's arrival rate: λ_6 (calls/hr)	41
Type VII call's arrival rate: λ_7 (calls/hr)	3
Type VIII call's arrival rate: λ_8 (calls/hr)	10
Type IX call's arrival rate: λ_9 (calls/hr)	1

With the same manner to generate inter-arrival time described in chapter 3, we continue to assume that the time between call arrivals is exponentially distributed.

Two simple data generation functions: 1) RANDBETWEEN ( ) in Microsoft Excel and 2) rexp ( ) in R were used to generate data sets. Table 5.2 summarizes the detail of how the data sets are generated.

Table 5.2: Data Generating Function

Parameter	Data Generating Function	Unit
Utilization rate	RANDBETWEEN (1,3) 1 for 85%, 2 for 90% , 3 for 100%	%
Inter-arrival time	rexp(*n*,*lamda*)	Seconds

Note: *n* is the number of observations, *lamda* is the desired call arrival rate.

With the above parameter setting and numerical data generation described above, a call center with the total of 30 inbound calls was generated with associated inter-arrival times and call types. The experiment compares the call center's performances among the different scheduling methods. The scheduling method is one of the mix approach (that is, MIXTFT, MIXMTF, and MIXMDCAW), and the current routing logic used at the actual call center (that is, WHAT IF). All of the optimization problems are solved on an INSPIRON B120 with a LINGO 10.0 commercial solver.

5.3.3 Discussion of Results

To measure the competency among the three mix approaches and the WHAT IF approach for inbound call scheduling in the case study call center, we conducted 1) descriptive statistics; and 2) one-tailed paired t-test statistics.

Descriptive Statistics

Tables 5.3 summarizes descriptive statistics of our findings.

Table 5.3: TFT – Descriptive Statistics

Unit: seconds	WHAT IF	MIXTFT	MIXMFT	MIXMDCAW
SumF	5364.6	3964.6	3895.6	3832.8
MaxF	258	186.8	188.8	127.76
MaxclDiv	159.34	198.3	245.81	148.2
Average F	178.82	124.36	129.85	148.2
Max W	45	94	115.6	10
Average W	5	17.27	18.31	1.04
Average solution time	approx. 0	4.7	3.8	approx. 0
% change of SumF with respect to WHAT IF	Base value	26%	27%	29%
% change of MaxF with respect to WHAT IF	Base value	26%	25%	49%
% change of MaxclDiv with respect to WHAT IF	Base value	24%	54%	7%
Generate better objective value than WHAT IF	Base value	Yes	Yes	yes
Frequencies of the occurrence: times (%)				
Feasible solutions found	30 (100%)	30 (100%)	30 (100%)	30 (100%)
No feasible solution found	0 (0%)	0 (0%)	0 (0%)	0 (0%)

Table 5.4: Descriptive Comparison of Solution Techniques

	Objective value		
	SumF	MaxF	MaxclDiv
Provide best objective value	MIXMDCAW	MIXMDCAW	MIXMDCAW
Provide better objective value compared to WHAT IF	MIXMDCAW, MIXMFT, MIXTFT	MIXMDCAW, MIXTFT, MIXMFT	MIXMDCAW
% change in objective value compare to WHAT IF	MIXMDCAW	MIXMDCAW	MIXMDCAW
Average speed of providing feasible scheduling solutions	WHAT IF, MIXMDCAW	WHAT IF, MIXMDCAW	WHAT IF, MIXMDCAW

From the descriptive statistics, we found that with the MIXTFT and MIXMFT, the call center experienced the higher call's waiting time. This mainly was due to the solution time of the optimization component of the approach. However this is well worth waiting since the better scheduling solutions can be found and as a consequence, the flow time can be reduced.

From our experiment, MIXMDCAW turned out to be a good approach in all performance measure. In this case study, the problem size in each optimization when we applied MIXMDCAW was so small that the solution time was close to zero second.

Due to the overall performances, we would recommend our MIXMDCAW approach for scheduling inbound calls in this call center as it usually provides good quality solution within tight time regardless of which objective is of interest.

<u>One-tailed Paired T-test Statistics</u>

Tables 5.5 and table 5.6 report the statistical comparison of solution techniques whether one solution technique can provide better call flow time and call waiting time than another at 95% confidence level. Note that =, <, or > symbols are provided to indicate when one solution technique significantly provided equal, lower, or higher mean value than the other at the 95% confidence level.

Table 5.5: Results of Paired T-test for Comparing Flow Time (F)

P(T<=t) one-tailed	Compare to			
	WATIF	MIXTFT	MIXMFT	MIXMDCAW
WHATIF	=	0.000 (>)	0.000 (>)	0.000 (>)
MIXTFT	0.000 (<)	=	0.356 (=)	0.330 (=)
MIXMFT	0.000 (<)	0.356 (=)	=	0.406 (=)
MIXMDCAW	0.000 (<)	0.330 (=)	0.406 (=)	=

Table 5.6: Results of Paired T-test for Comparing Waiting Time (W)

P(T<=t) one-tailed	Compare to			
	WATIF	MIXMDCAW	MIXTFT	MIXMFT
WHATIF	=	0.942 (=)	0.014 (<)	0.038 (<)
MIXMDCAW	0.942 (=)	=	0.356 (=)	0.006 (<)
MIXTFT	0.014 (>)	0.356 (=)	=	0.562 (=)
MIXMFT	0.038 (>)	0.006 (>)	0.562 (=)	=

Summary of findings:

One-tailed paired T-test shows at 95% confidence level:

1. Comparison of F values
   - MIXTFT, MIXMFT, and MIXMDCAW significantly provide the scheduling solutions with lower F values than WHAT IF approach.

- There is no significant difference among F values obtained from MIXTFT, MIXMFT, and MIXMDCAW.

2. Comparison of W values
    - WHAT IF approach significantly provides the scheduling solution with lower W values than MIXMFT and MIXTFT.
    - MIXMDCAW significantly provides the scheduling solution with lower W values than MIXMFT.
    - There is no significant difference among W values obtained from MIXMDCAW and WHAT IF.
    - There is no significant difference among W values obtained from MIXMDCAW and MIXTFT.
    - There is no significant difference among W values obtained from MIXTFT and MIXMFT.

3. There is no significant difference among W value obtained from
    - MIXMDCAW and WHAT IF.
    - MIXMDCAW and MIXTFT.
    - MIXTFT and MIXMFT.

4. From 1 to 3, we can conclude at 95% confidence level that

For F value: WHAT IF>MIXTFT=MIXMFT=MIXMDCAW
For W value:

WHAT IF = MIXMDCAW < MIXMFT
WHAT IF < MIXTFT
MIXMDCAW = MIXTFT
MIXTFT = MIXMFT

Our findings from the t-test show that MIXMDCAW is the most preferable method to significantly reducing call flow time while keeping the waiting time unchanged. However, MIXTFT and MIXMFT are still a preferable method to the current inbound calls routing logic used by the center since it significantly reduce the mean flow time.

5.4 Conclusion of Case Study

In this chapter, we suggested a composite method among cutting plane algorithm, FCFS and HEU method to help call center improve its performances. The results indicated that the suggested solution approach can offer effective solution techniques for inbound call scheduling problems under tight time constraint. This finding confirms the benefit of using the optimization approach for real time scheduling of inbound calls.

CHAPTER 6. CONCLUSION AND FUTURE RESEARCH

6.1 Conclusion

The purpose of our study has been to identify an effective approach for scheduling inbound calls in call centers. To achieve the objective, we have developed three different Integer Programming (IP) problems for inbound call scheduling, with objective functions of 1) minimizing the Total Flow Time (TFT), 2) minimizing the Maximum Flow Time (MFT) of any call, and 3) minimizing the Maximum Deviation of Cumulative Assigned Workload (MDCAW) for CSRs.

The performance of those developed IPs were demonstrated through numerical examples. The experiment is designed to be realistic based on the model provided by the actual call center. In our study, three cases of skill distribution are considered in the experiment: In Case I, each CSR is skilled to work with only one type of call. In Case II, there are identical CSRs that skilled to work with any types of call. In Case III, CSRs are skilled to work with any types of call but at the different service rate. The results of numerical experiment evaluates under what conditions these IP formulations give superior performance and which objective should be chosen. Results indicate that optimization compares favorably to FCFS under realistic scenarios involving specialized but broadly trained CSRs and high call center utilization rates.

While the performance of call centers can be improved by employing an optimization approach for scheduling inbound calls, the long computation times could limit it use in a real-time ACD system. Four solution techniques consisting of IP reformulation, Lagrangian relaxation and duality, cutting plane algorithm, and heuristic optimization approach are purposed for solving the formulated IPs. There are therefore twelve new IPs: 1) Reformulated problem TFT (RTFT), 2) Reformulated problem MFT (RMFT), 3) Reformulated problem MDCAW (RMDCAW), 4) Lagrangian Dual of problem TFT (LDTFT), 5) Lagrangian Dual of problem MFT (LDMFT), 6) Lagrangian Dual of problem MDCAW (LDMDCAW), 7) Cutting Plan algorithm of problem TFT (CTFT), 8) Cutting

Plan algorithm of problem MFT (CMFT), 9) Cutting Plan algorithm of problem MDCAW (CMDCAW), 10) Heuristic Optimization approach of problem TFT (HEUTFT), 11) Heuristic Optimization approach of problem MFT (HEUMFT), and 12) Heuristic Optimization approach of problem MDCAW (HEUMDCAW).

In exploring the feasibility of the purposed solution techniques, we adopt a preliminary experiment. In this experiment, only the IPs with the total flow time objective function are investigated to evaluate the feasibility of the purposed solution techniques. We also developed Lagrangian relaxation with respect to the sequencing constraints of problem TFT (LRTFT) to obtain the lower bound values of problem LDTFT. The quality of bound generating by the continuous relaxation (CR) of RTFT, LRTFT when Lagrangian multiplier (λ) is set to be 0, the CR of CTFT, and HEUTFT versus the CR of TFT under a tight time constraint were explored and compared. The preliminary indicates that Reformulation techniques consumed long solution time and provided weak bound value. Hence, all of the purposed solution techniques, except for the IP reformulation technique, were very promising and were included in our further examination

The second phase (the actual experiment) was set up to examine the effect of Lagrangian Relaxation and Duality, Cutting Plane Algorithm, and Heuristic Optimization approach. In addition to these promising solution techniques, we also included HEU method, the logic of obtaining heuristic feasible scheduling solutions, as another solution technique in our focus. To signify the effective approaches for scheduling inbound calls, the objective values and the computational times of solving the IPs using a standard solver were compared. Results indicate that Heuristic Optimization approach is the most preferable when objective is to minimize the total flow time. When objective is to minimize the maximum flow time or to minimize the maximum deviation of cumulative assigned workload, Lagrangian Relaxation and Duality could be as preferable as Heuristic Optimization approach if we can improve the ability of finding feasible scheduling solution. However if solution time is a concern to call centers, Cutting Plane Algorithm is the next preferable method.

A case study of a call center was also conducted to illustrate the use of Optimization approach for real time scheduling of inbound calls. In this study, we suggested a composite method among cutting plane algorithm, FCFS and HEU method to help the modeled call center improve its performances. Three new mix approach (that is, MIXTFT, MIXMFT, and MIXMDCAW) were introduced. We then constructed a numerical experiment to compare the call center performances resulted from using the suggested solution techniques with its current inbound call scheduling method (referred as WHAT IF method). The descriptive statistics and t-test results indicated that the suggested solution approaches is preferable to the WHAT IF method since it can offer effective solution techniques for inbound call scheduling problems under tight time constraint.

In conclusion, this study shows that the performance of call centers can be improved by employing an optimization approach for inbound call scheduling. With effective solution techniques call center manager can obtain good schedule quality within reasonably solution times.

6.2 Future Research

1. In this study we have shown that our purpose scheduling methods are very effective for scheduling inbound calls in busy small-sized call centers (less than 10 CSRs). We believe that the same techniques can be used in bigger-sized call centers as well. One possibility is to divide the call centers in to smaller groups so that the techniques can be implemented to find scheduling solutions.

2. Although our reformulation IPs are not effective, the other approach, such as set covering problem, might be more suitable for reformulate the inbound calls scheduling problem.

3. There are several methods for solving Lagrangian dual problems. In our study, we used a simple subgradient algorithm to solve the problem. We found that the quality of the solutions is superior to those obtained from FCFS; however, on occasion feasible solutions could not be obtained due to time limitation. Applying more powerful solution techniques for solving Lagrangian dual might be more appropriate.

BIBLIOGRAPHY

Adel, G. and Pearce, W. (1979) 'Telephone sales manpower planning at Qantas', *Interface*, Vol. 9, No. 3, pp.1-9.plane methods in service systems', *Proceedings of the 2002 National Science Foundation Design, Service and Manufacturing Grantees Conference*, http://legacy.orie.cornell.edu/~shane/pubs/NSFCutting.pdf

Anderson, E.D. and Anderson, K.D. (1995) 'Presolving in linear programming', *Mathematical Programming*, Vol. 71, pp. 221-245.

Barton, K. (2004) 'Call center analysis using simulation modeling', IE599 project report, Iowa State University.

Berman, O. and Larson, R.C. (1994) 'Determining optimal pool size of a temporary call-in work force', *European Journal of Operations Research*, Vol. 73, pp.55–64.

Bertsimas, D. and Tsitsiklis, J.N. (1997) *Introduction to Linear Optimization*, Athena Scientific, Belmont, Massachusetts.

Borst, S.C. (1995) 'Optimal probabilistic allocation of customer types to servers', *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS95)*, Ottawa, ON, Canada, pp.116–125.

Brearley, A.L., Mitra, G. and Williams, H.P. (1975) 'Analysis of mathematical programming problems prior to applying the simplex method', *Mathematical Programming*, Vol. 8, pp. 54-53.

Balas, E. (1975a) 'Facets of the knapsack polytope', *Mathematical Programming*, Vol. 8, pp. 146-164.

Barnhart, C., Johnson, E.L., Nemhauser, G.L., Savelsbergh, M.W.P., and Vance, P.H. (1998) 'Branch-and-Price: Column Generation for Huge Integer Programs', *Operations Research* Vol. 46, pp. 316-329.

Bruce, A. and Parson, H. (1993) 'Establishing telephone-agent staffing level through economic optimization', *Interfaces*, Vol. 23, No. 2, pp.14–20.

Cezik, T., Gunluk, O. and Luss, H. (2001) 'An Integer Programming model for the weekly tour scheduling problem', *Naval Research Logistics*, Vol. 48, No. 7, pp.607–624.

Cho, D.C., Johnson, E. L., Padberg, M. W. and Rao, M. R. (1983a) 'On the uncapacitated plant location problem I: valid inequalities and facets', *Mathematics of Operations Research,* Vol. 8, pp. 579-589.

Cho, D.C., Padberg, M. W. and Rao, M. R. (1983b) 'On the uncapacitated plant location problem II: facets and lifting theorems', *Mathematics of Operations Research,* Vol. 8, pp. 590-612.

Clausen, J. (1999) 'Branch and bound algorithms principles and examples', Department of computer science, University of Copenhagen, Denmark, http://www.imada.sdu.dk/Courses/DM85/TSPtext.pdf

Climer, S. and Zhang, W. (2006) 'Cut-and-solve: An iterative search strategy for combinatorial optimization problem', *Artificial Intelligence*, Vol. 170, pp. 714-738

Durinovic, J.S. and Levy, Y. (1997) 'Advanced routing solutions for toll-free customers: algorithm design and performance', *International Teletraffic Congress (ITC-15)*, Amsterdam, pp.157–167.

De Farias, I.R., Jr. and Nemhauser, G.L. (2001) 'A family of inequalities for the generalized assignment polytope', *Operations Research Letters*, Vol. 29, pp. 49–51.

Feinberg, M.A. (1990) 'Performance characteristics of automated call distribution systems', *Global Telecommunications Conference (GLOBECOM)*, New York, pp.415–419.

Gans, N. and Zhou, Y. (2003) 'A call-routing problem with service-level constraints', *Operations Research*, Vol. 51, No. 2, pp.255–271.

Gans, N., Koole, G. and Mandelbaum, A. (2003) 'Telephone call centers: tutorial, review, and research prospects', *Manufacturing and Service Operations Management*, Vol. 5, pp.79–141.

Garnett, O. and Mandelbaum (2000), *An Introduction to Skills-Based Routing and its Operational Complexities*, Teaching note, Technion, http://fic.wharton.upenn.edu/fic/f0503mandelbaum.pdf

Geoffrion, A.M., Marsten, R.E. (1972) 'Integer Programming algorithms: a framework and state-of-the-art survey', *Management Science*, Vol. 18, pp.465-491.

Gomory, R.E. (1958) 'Outline of an algorithm for integer solution to linear programs', *Bulletin American Mathematical Society,* Vol. 64, pp.275-278.

Gottfried, B.S. and Weisman, J. (1973) *Introduction to Optimization Theory*, Prentice Hall, Englewood Cliffs, New Jersey.

Gottlieb, E.S. and Rao, M.R. (1986) 'The generalized assignment problem II: three classes of facets', Technical Report #8622-Stat, Baruch College, The City University of New York.

Gottlieb, E.S. and Rao, M.R. (1990) 'The generalized assignment problem: valid inequalities and facets. *Mathematical Programming*, Vol. 46, pp. 31-52.

Grotschel, M. and Padberg, M.W. (1979) 'On the symmetric traveling salesman problem I: inequalities', *Mathematical Programming*, Vol. 16, pp. 265-280.

Guignard, M. and Spielberg, K. (1981) 'Logical reduction methods in zero-one programming: minimal preferred inequalities', *Operations Research*, Vol. 29, pp. 49-74

Gulati, S. and Malcolm, S. (2001) 'Call center scheduling technology evaluation using simulation', *Winter Simulation Conference*, pp.1438–1442.

Hammer, P.L., Johnson, E.L and Peled, U.N. (1975) 'Facets of regular 0-1 polytopes', *Mathematical Programming*, Vol. 8, pp. 179-206.

Henderson, W.B. and Berry, W.L. (1976) 'Heuristic methods for telephone operator shift scheduling: an experimental analysis', *Management Science*, Vol. 22, No. 12, pp.1372–1380.

Hoffman, K.L. and Padberg, M. (1985) 'LP-based combinatorial problem solving', *Annals Operations Research*, Vol. 4, pp.145-194.

Kogan, Y., Levy, Y. and Milito, R.A. (1997) 'Call routing to distributed queues: is FIFO really better than MED?', *Telecommunication Systems – Modeling, Analysis, Design and Management*, Vol. 7, Nos. 1–3, pp.299–312.

Koole, G. and Mandelbaum, A. (2002) 'Queueing models of call centers: an introduction', *Annals of Operations Research*, Vol. 113, pp.41–59.

Koole, G., Pot, A. and Talim, J. (2003) 'Routing heuristics for multi-skill call centers', *Proceeding of the 2001 Winter Simulation Conference Winter Simulation Conference*, pp.1813–1816.

Land, A.H. and Doig, A.G. (1960) ‘An automatic method of solving discrete programming problems’, *Econometrica*, Vol. 28, pp.497-520.

Leung, J.M.Y. and Magnanti, T. L. (1986) ‘Valid inequalities and facets of the capacitated plant location problem’, Working Paper No. OR 149-86, Operations Research Center, MIT.

Levy, Y., Durinovic, J.S. and Milito, R.A. (1994) ‘Dynamic network call distribution with periodic updates’, *International Teletraffic Congress (ITC-14)*, Amsterdam, pp.85–94.

Mandelbaum, A. (2004) *Call Centers (Centres): Research Bibliography with Abstracts*, Version 3, p.201.

Marbach, P., Mihatsch, O. and Tsitsiklis, J.N. (1998) ‘Call admission control and routing in integrated services networks using reinforcement learning’, *IEEE Conference on Decision and Control*, IEEE, Piscataway, NJ, pp.563–568.

Mason, A.J., Ryan, D.M. and Panton, D.M. (1998) ‘Integrated simulation, heuristic and optimization approaches to staff scheduling’, *Operations Research*, Vol. 46, No. 2, pp.161–175.

Melsa, P.J.W., Kenney, J.B. and Rohrs, C.E. (1990) ‘A neural network solution for call routing with preferential call placement’, *Global Telecommunications Conference (GLOBECOM)*, San Diego, CA, pp.1377–1381.

Mitchell, J.E. (2002) ‘Branch-and-cut algorithms for combinatorial optimization problems’, Handbook of Applied Optimization, pp. 65-77, Oxford University Press, January 2002.

Nemhauser, G.L. and Wolsey, L.A. (1988) *Integer and Combinatorial Optimization*, John Wiley, New York.

Pinedo, M., Seshadri, S. and Shanthikumar, J.G. (2000) ‘Call centers in financial services: strategies, technologies, and operations’, In E.L. Melnick, P. Nayyar, M.L. Pinedo, and S. Seshadri, editors, *Creating Value in Financial Services: Strategies, Operations and Technologies*, Kluwer Academic Pubishers, Boston, MA. pp. 357-388.

Pochet. Y. (1988) ‘Valid inequalities and separation for capacitated economic lot-sizing’, *Operations Research Letters*, Vol. 7, pp.109-116.

Sharp, D.E. (2003) *Call Center Operation: Design, Operation, and Maintenance*, Elsevier Science, Burlington, MA.

Thompson, G.M. (1995) ‘Improved implicit optimal modeling of the labor shift scheduling problem’, *Management Science*, Vol. 41, No. 4, pp.595–607.

Vuthipadadon, S. and Olafsson, S. (2007) ‘An integer programming approach for scheduling inbound calls in call centres’, *International Journal of Operation Research*, Vol. 2, No.4, pp. 414-428.

Williams, H.P. (1985). *Model Building in Mathematical Programming,* 2nd ed. Wiley, New York.

Wolsey, L.A. (1975) ‘Faces for a linear inequality in 0-1 variables’, *Mathematical Programming*, Vol. 8, pp. 165-178.

APPENDIX A. THE COMPLETE LIST OF NOTATION

This appendix shows the completed list of notation using in this research paper.

Variable Name	Description
$ACL_j(t)$	Actual cumulative work load of CSR j calculated at time t
AvgF	Average flow time of a call
AvgN	Average number of calls in system
AvgW	Average waiting time for a call
$CL_j(t)$	cumulative assigned work load of CSR j calculated at time t
$F_i(t)$	flow time of call i at time t
I	Set of calls
J	Set of CSRs
$J_i(t)$	CSR which is assigned at time t to serve call i at time t
L	Load balance factor
$m(t)$	Number of calls in the call center at time t waiting to be service
Maxcldiv	Maximum deviation of assigned workload
MaxF	Maximum flow time of a call
MaxN	Maximum number of calls
MaxW	Maximum waiting time for a call
$n(t)$	Number of CSRs in the call center at time t
P_{ij}	Server dependent processing time of call i served by CSR j
Q	Set of queue positions
$Q_i(t)$	Queue position of call i at time t
$QL_i(t)$	Queue length or the total processing time spent by a CSR to complete calls prior than call i at time t
$r_j(t)$.	Remaining serving time of CSR j at time t
$SA_i(t)$	Speed answer or the total time it takes to answer call I at time t
$s_j(t)$	Number of calls that CSR j started the service after time t and before t_1
SL	Call center service level, which is usually defined as some fixed Z percent of calls answered in Y seconds.
t	Time period that call center performs the optimization to scheduling inbound calls
t_1	Time period that call center will perform the next optimization to scheduling inbound calls
t_0	Time period that call center previously performs optimization to scheduling inbound calls
$U_{ij}(t)$	An inbound call scheduling decision at time t, which equals 1 if call i is assigned to CSR j, and equals 0 otherwise.
$V_{iq}(t)$	An inbound call scheduling decision at time t, which equals 1 if call i is sequenced to be at queue position q, and equals 0 otherwise.
$w_i(t)$	Waiting time of call i at time t
$X_{ijq}(t)$	An inbound call scheduling decision at time t, which equals 1 if call i is assigned to CSR j at queue position q, and equals 0 otherwise.
$\overline{X}^{FCFS}$	Scheduling solutions from FCFS
$\overline{X}^{1}$	Scheduling solutions from the previous optimization
$\overline{X}^{1'}$	Heuristic feasible scheduling solutions
$\overline{X}^{1*}$	Initial feasible scheduling solutions
$\overline{X}^{2}$	New scheduling solutions deriving from the proposed heuristic approach
$\chi_{\{f(t)\leq g(t)\}}$	Counting index equals 1 if f(x)≤g(x) and equals 0 otherwise.

APPENDIX B. PRELIMINARY EXPERIMENT

To explore the feasibility of the purposed solution techniques, we adopt a preliminary experiment. In experiment, we vary the number of calls and number of CSRs in a setting in a range from 4 to 9. There are therefore 36 different settings. For every setting, two number of time (in second) are randomly generated. The smaller number, at 250 seconds, is used as the time period that call center performs the optimization to scheduling inbound call (t). The bigger number, at 454 seconds, is used as the time period that call center will perform the next optimization to scheduling inbound calls. Therefore the time horizon considered t- t_1 is set to 204. We also specify the call center's parameter settings using the randomly simulated preliminary experiment's data set as shown in Table B.1, Table B.2 and Table B.3.

Table B.1: Call type and its waiting time $w_i(t)$

Call i	1	2	3	4	5	6	7	8	9
Type	1	2	1	3	1	1	2	3	1
$w_i(t)$	45	32	30	30	27	14	7	3	1

Table B.2 CSR Actual cumulative workload $Acl_j(t_0)$ and remaining serving time $r_j(t)$

CSR j	1	2	3	4	5	6	7	8	9
$Acl_j(t_0)$	12	35	54	45	0	16	32	3	34
$r_j(t)$	132	114	56	233	322	130	345	34	67

Table B.3 Server dependent processing time P_{ij}

CSR j	1	2	3	4	5	6	7	8	9
Call Type									
1	15	56	12	45	130	42	234	45	87
2	50	12	78	45	24	89	200	69	87
3	23	25	12	20	65	42	134	167	145

In the preliminary experiment, we attempt to explore and compare the solution time and the quality of bound generating by the continuous relaxation (CR) of RTFT, LDTFT, the CR of CTFT, and HEUTFT versus the CR of TFT under a tight time constraint which is set, as an example, to be 180 seconds. However, with the time limitation, it is not possible for us to develop the programming code for solving Lagrangian Dual. Therefore in our preliminary experiment; we develop the programming code for solving Lagrangian relaxation with respect to the sequencing constraints of problem TFT to obtain the lower bound values of problem LDTFT. For more simplicity, we specify the Lagrangian multiplier (λ) to be 0, and refer the problem as LRTFT. As a consequence, we can only explore and compare the solution time and the quality of bound generating by the CR of RTFT, LRTFT, the CR of CTFT, and HEUTFT versus the CR of TFT under a tight time constraint which is set to be 180 seconds. There are therefore 180 scenarios.

Under each scenarios, the solution time, number of iteration, and the objective value obtain are observed and compared. In this experiment, the reason we only work with the total flow time objective function is that we believe with only an objective function we have some idea of the feasibility of the purposed solution techniques.

We use an INSPIRON B120 to run our experiment with LINGO 10.0 as an optimization solver. We found that:

1) In 3 of 36 experiments using the CR of RTFT, and 1 of 36 experiments using LRTFT could not provide the solution within 180 sec, while all of the experiments using CTFT and TFT can provide the solution for the CR problem. HEUTFT always provides solutions. Hence, among the proposed technique solutions CTFT, LRTFT, HEUTFT are promising approaches to explore for the total flow time objective.
2) In 8 of 33 experiments using RTFT, 35 of 36 experiments using CTFT, the solution for the CR problem are also the IP solutions, while only 15 of 36 experiments using the CR of TFT can provide the IP solution. HEUTFT and LRTFT always provide integer solutions. Hence, CTFT is a strong approach to provide an upper bound and HEUTFT is a strong approach to provide an initial feasible solution for the total flow time objective.
3) The one-way ANOVA test between the CR of RTFT, LRTFT, the CR of CTFT, and HEUTFT versus the CR of TFT are performed for solution time, number of iteration and

the objective value to determine the different between approaches. Their average value and p-value are given table B.4

Table B.4 One-way ANOVA test

CR of TFT VS	Solution time (sec)			# of Iteration			Objective value (sec)		
	TFT Avg	Avg	P-Value	TFT Avg	Avg	P-Value	TFT Avg	Avg	P-Value
CR of RTFT	17.11	98.42	1.3E-06	15399.14	49833.58	8.3E-04	566.55	399.48	1.1E-03
CR of CTFT		4	4.7E-02		1491.06	4.3E-02		595.72	6.0E-01
LRTFT		3.26	3.8E-02		1650.06	4.6E-02		51023	2.6E-01
HEUTFT		0	9.4E-02		0	2.4E-02		853.97	4.8E-04

ANOVA test indicates that:

- There are significantly differences in solution time and number of iteration to find the solution between all techniques versus CR of TFT with 99% confidence. However, RTFT tends to use more solution time and have higher number of iteration than every approach to obtain bound value. Hence, all the proposed solution techniques, except RTFT, are very strong approaches to decrease solution time, and number of iteration.
- There are significantly differences in objective value between CR of RTFT, and HEUTFT versus CR of TFT with 99% confidence, between LRTFT versus CR of TFT with 95% confidence. However for CR of CTFT, such confidence cannot be established. We found that the CR of RTFT tends to provide weaker lower bound than the CR of TFT. For LRTFT, although we found that its lower bound quality obtained is slightly worst than CR of TFT, its solution time does makes it a promising approach. In addition, since LRTFT only provides a lower bound value to LDTFT, and in theory, the Lagrangian dual is a stronger relaxation than the continuous relaxation, hence it is still worth to continue exploring the LDTFT technique.
- In term of quality of the upper bound CTFT is a better approach. This is as our expectation since the objective value from HEUTFT is not actual upper bound. However it is still benefit to use HEUTFT approach to provide an initial solution to the optimization.

- Table B.5 provides p-value of a one-way ANOVA test between CR of CTFT versus HEUTFT. From the one-way ANOVA test, we can say with 99% confidence that CR of CTFT and HEUTFT are different in solution time, number of iteration and objective value. This mean CTFT and HEUTFT are competitive approaches since although we found CTFT is good to provide upper bound value but it takes more solution time and higher number of iteration to solve for the solution.

Table B.5 One-way ANOVA test between CTFT and HEUTFT

	P-value		
	Solution time (sec)	# of Iteration	Objective value (sec)
CR of CTFT VS HEUTFT	3E-05	1E-03	1E-03

In conclusion of the preliminary result, we believe that all of the purposed solution techniques, except for the IP reformulation technique, are very promising and worth continuing examination

APPENDIX C. DATA SETS USED IN SOLUTION TECHNIQUE EXPERIMENTS

Table C.1: Call arrival rate λ, number of CSRs *n(t)*, number of calls *m(t)*

Case	λ	*n(t)*	*m(t)*			Case	λ	*n(t)*	*m(t)*		
			TFT	MFT	MDCAW				TFT	MFT	MDCAW
1	18	4	3	3	8	16	22	4	3	3	7
2	32	5	3	2	5	17	15	7	4	4	4
3	19	6	3	2	7	18	25	8	2	2	2
4	17	5	3	3	3	19	12	6	3	3	3
5	12	3	5	4	4	20	18	4	3	3	5
6	26	9	2	2	2	21	17	6	3	3	4
7	23	5	3	2	2	22	28	9	3	3	3
8	9	4	4	4	4	23	14	7	2	2	2
9	21	6	4	4	4	24	16	6	3	3	4
10	31	9	3	2	2	25	21	8	3	2	2
11	23	6	2	2	2	26	11	4	2	3	6
12	20	7	3	3	3	27	22	7	2	2	2
13	11	4	2	2	2	28	16	4	6	4	6
14	12	5	3	3	3	29	14	5	2	2	3
15	23	7	2	2	2	30	20	6	2	3	2

Table C.2: Server dependent processing time P_{ij}

Case	Call Type	CSR *j*								
		1	2	3	4	5	6	7	8	9
1	1	101	557	789	78					
	2	286	422	193	1159					
	3	2023	566	1379	1019					
2	1	317	287	682	130	248				
	2	317	278	555	1353	4678				
	3	230	122	72	255	154				
3	1	716	228	399	402	991	1496			
	2	696	153	2332	580	261	169			
	3	114	140	2375	444	949	1345			
4	1	1080	498	69	1478	743				
	2	1504	1125	4084	1567	1989				
	3	33	93	270	25	6				
5	1	789	692	739						
	2	183	524	65						
	3	520	192	910						
6	1	2143	981	151	1591	1636	670	616	110	722
	2	1796	574	164	2730	134	402	1621	2308	3990
	3	46	494	514	973	730	1205	696	13	559
7	1	1060	551	1587	140	892				
	2	1542	1164	306	123	778				
	3	1850	99	182	103	368				

Table C.2: Server dependent processing time P_{ij} (Cont.)

Case	Call Type	CSR *j*								
		1	2	3	4	5	6	7	8	9
8	1	481	4537	146	357					
	2	668	1351	3251	1078					
	3	1453	185	3461	1697					
9	1	609	676	1455	1101	1049	5347			
	2	1309	2040	3728	3845	1657	339			
	3	51	17	31	34	89	472			
10	1	203	257	612	605	142	78	894	248	174
	2	1664	562	1913	1637	853	145	442	1607	346
	3	1501	763	1487	461	2522	971	3827	175	1690
11	1	441	1705	268	1110	404	827			
	2	221	23	117	31	153	739			
	3	935	826	27	2002	1097	355			
12	1	1955	2008	3250	1096	1830	934	1838		
	2	1754	4054	69	975	106	728	1966		
	3	19	217	589	419	99	1192	443		
13	1	3550	191	26	1349					
	2	2089	2858	516	755					
	3	490	230	396	834					
14	1	118	2997	1246	537	2532				
	2	160	547	531	127	1113				
	3	348	146	992	289	2401				
15	1	320	1599	1428	149	1341	269	899		
	2	104	573	1963	326	296	1474	61		
	3	4140	583	969	677	870	1475	1395		
16	1	79	26	1389	107					
	2	3536	689	108	2527					
	3	97	62	159	325					
17	1	104	1218	240	607	1295	282	1330		
	2	1184	2284	337	1484	4615	1180	1991		
	3	3496	118	4437	444	3564	597	251		
18	1	236	623	362	198	1260	3006	163	1465	
	2	2486	146	3115	206	80	2591	804	376	
	3	567	681	1345	123	198	403	3006	232	
19	1	209	1153	5298	2881	1727	200			
	2	947	1333	3728	1020	308	215			
	3	3103	4959	1818	1723	1935	319			
20	1	793	665	1134	124					
	2	1401	2396	184	219					
	3	51	1804	279	141					
21	1	1241	271	30	192	912	923			
	2	2402	145	1388	4018	3780	710			
	3	17	729	17	27	180	236			
22	1	213	106	517	229	70	142	145	564	82
	2	768	1866	689	212	533	1059	13781	541	556
	3	2005	567	1364	273	341	1263	1756	1062	998
23	1	559	180	295	227	1022	1219	2860		
	2	1874	2594	673	2836	157	857	1157		
	3	632	1417	1415	1666	1403	3431	39		

Table C.2: Server dependent processing time P_{ij} (Cont.)

Case	Call Type	CSR *j*								
		1	2	3	4	5	6	7	8	9
24	1	456	1311	4270	456	2582	4284			
	2	1292	499	970	930	1206	3925			
	3	357	161	141	433	333	100			
25	1	2446	5641	694	1420	815	309	236	1206	
	2	1080	1844	2369	1731	82	453	30	1723	
	3	1171	1208	86	3084	374	999	719	881	
26	1	568	1188	39	1169					
	2	171	819	158	345					
	3	544	1818	1414	3720					
27	1	18	1233	34	740	24	1713	1046		
	2	2838	2349	39	2008	742	2269	2469		
	3	337	79	708	138	2734	1068	132		
28	1	1081	15	2416	1166					
	2	31	1082	290	591					
	3	35	205	234	570					
29	1	1826	223	868	287	3132				
	2	517	246	391	354	135				
	3	1766	869	1593	1855	621				
30	1	2161	1021	1235	1291	152	268			
	2	140	136	444	179	121	149			
	3	1719	77	29	1076	1191	3358			

APPENDIX D. ONE-TAILED PAIR T-TEST

Note that =, <, or > symbols are provided to indicate when one solution technique significantly provided equal, lower, or higher mean value than the other at the 95% confidence level.

Table D.1: TFT – Results of Paired T-test for Comparing Mean of Total Flow Time

P(T<=t) one-tailed	Compare to					
	FCFS	HEU	LDTFT	TFT	CTFT	HEUTFT
FCFS	=	0.000 (>)	0.000 (>)	0.000 (>)	0.000 (>)	0.000 (>)
HEU	0.000 (<)	=	0.045 (>)	0.035 (>)	0.012 (>)	0.001 (>)
LDTFT	0.000 (<)	0.045 (<)	=	0.472 (=)	0.286 (=)	0.019 (>)
TFT	0.000 (<)	0.035 (<)	0.472 (=)	=	0.221 (=)	0.004 (>)
CTFT	0.000 (<)	0.012 (<)	0.286 (=)	0.221 (=)	=	0.001 (>)
HEUTFT	0.000 (<)	0.001 (<)	0.019 (<)	0.004 (<)	0.001 (<)	=

Table D.2: TFT – Results of Paired T-test for Comparing Mean of Solution Time

P(T<=t) one-tailed	Compare to						
	LDTFT Total time	TFT	LDTFT best time	HEUTFT	CTFT	HEU	FCFS
LDTFT total time	=	0.020 (>)	0.000 (>)	0.010 (>)	0.000 (>)	0.000 (>)	0.000 (>)
TFT	0.020 (<)	=	0.368 (=)	0.438 (=)	0.000 (>)	0.000 (>)	0.000 (>)
LDTFT best time	0.006 (<)	0.368 (=)	=	0.240 (=)	0.001(>)	0.000 (>)	0.000 (>)
HEUTFT	0.010 (<)	0.438 (=)	0.240 (=)	=	0.000 (>)	0.000 (>)	0.000 (>)
CTFT	0.000 (<)	0.000 (<)	0.001 (<)	0.000 (<)	=	0.000 (>)	0.000 (>)
HEU	0.000 (<)	0.000 (<)	0.000 (<)	0.000 (<)	0.000 (<)	=	=
FCFS	0.000 (<)	0.000 (<)	0.000 (<)	0.000 (<)	0.000 (<)	=	=

Table D.3: MFT – Results of Paired T-test for Comparing Mean of Max Flow Time

P(T<=t) one-tailed	Compare to					
	FCFS	HEU	MFT	CMFT	LDMFT	HEUMFT
FCFS	=	0.000 (>)	0.000 (>)	0.000 (>)	0.000 (>)	0.000 (>)
HEU	0.000 (<)	=	0.033 (>)	0.026 (>)	0.003 (>)	0.002 (>)
MFT	0.000 (<)	0.033 (<)	=	0.358 (=)	0.047 (>)	0.015 (>)
CMFT	0.000 (<)	0.026 (<)	0.358 (=)	=	0.011 (>)	0.001 (>)
LDMFT	0.000 (<)	0.003 (<)	0.047 (<)	0.011 (<)	=	0.405 (=)
HEUMFT	0.000 (<)	0.002 (<)	0.015 (<)	0.001 (<)	0.405 (=)	=

Table D.4: MFT – Results of Paired T-test for Comparing Mean of Solution Time

P(T<=t) one-tailed	Compare to						
	LDMFT total time	MFT	LDMFT best time	HEUMFT	CMFT	HEU	FCFS
LDMFT total time	=	0.001 (>)	0.002 (>)	0.008 (>)	0.000 (>)	0.000 (>)	0.000 (>)
MFT	0.001 (<)	=	0.212 (=)	0.187 (=)	0.005 (>)	0.003 (>)	0.003 (>)
LDMFT best time	0.002 (<)	0.212 (=)	=	0.222 (=)	0.013 (>)	0.008 (>)	0.008 (>)
HEUMFT	0.008 (<)	0.187 (=)	0.222 (=)	=	0.001 (>)	0.000 (>)	0.000 (>)
CMFT	0.000 (<)	0.005 (<)	0.013 (<)	0.001 (<)	=	0.000 (>)	0.000 (>)
HEU	0.000 (<)	0.003 (<)	0.008 (<)	0.000 (<)	0.000 (<)	=	=
FCFS	0.000 (<)	0.003 (<)	0.008 (<)	0.000 (<)	0.000 (<)	=	=

Table D.5: MDCAW – Results of Paired T-test for Comparing Mean of Max Deviation of Cumulative Assigned Workload

P(T<=t) one-tailed	Compare to					
	FCFS	HEU	MDCAW	CMDCAW	LDMDCAW	HEUMDCAW
FCFS	=	0.007 (>)	0.002 (>)	0.000 (>)	0.006 (>)	0.000 (>)
HEU	0.007 (<)	=	0.154 (=)	0.011 (>)	0.029 (>)	0.001 (>)
MDCAW	0.002 (<)	0.154 (=)	=	0.108 (=)	0.235 (=)	0.007 (>)
CMDCAW	0.000 (<)	0.011 (<)	0.108 (=)	=	0.196 (=)	0.006 (>)
LDMDCAW	0.006 (<)	0.029 (<)	0.235 (=)	0.196 (=)	=	0.081 (=)
HEUMDCAW	0.000 (<)	0.001 (<)	0.007 (<)	0.006 (<)	0.081 (=)	=

Table D.6: MDCAW – Results of Paired T-test for Comparing Mean of Solution Time

P(T<=t) one-tailed	Compare to						
	MDCAW	LDMDCAW total time	LDMDCAW best time	HEU MDCAW	CMDCAW	HEU	FCFS
MDCAW	=	0.131 (=)	0.296 (=)	0.003 (>)	0.000 (>)	0.000 (>)	0.000 (>)
LDMDCAW Total time	0.131 (=)	=	0.003 (>)	0.069 (=)	0.004 (>)	0.004 (>)	0.004 (>)
LDMDCAW Best time	0.296 (=)	0.003 (<)	=	0.120 (=)	0.008 (>)	0.007 (>)	0.007 (>)
HEUMDCAW	0.003 (<)	0.069 (=)	0.120 (=)	=	0.000 (>)	0.000 (>)	0.000 (>)
CMDCAW	0.000 (<)	0.004 (<)	0.008 (<)	0.000 (<)	=	0.001 (>)	0.001 (>)
HEU	0.000 (<)	0.004 (<)	0.007 (<)	0.000 (<)	0.001 (<)	=	=
FCFS	0.000 (<)	0.004 (<)	0.007 (<)	0.000 (<)	0.001 (<)	=	=

APPENDIX E. DATA SETS USED IN CASE STUDY

Table E.1: CSRs' skill set

CSR/Call Type	1	2	3	4	5	6	7	8	9
1	4						4		
2				4	4				4
3		2		2	1	3			
4	1				3		1		
5				2	1	1			1
6				1	1				1
7		2		2		1			
8	1				3		1		
9				2	1				2
10				2	2	1			2
11	3		3						
12	3			2		1			
13		2		2	1	1			2
14	3				1		3		2
15	1						1		
16	1	3			3	3	1		
17				2	2	1			2
18	1	3			3	3			
19	3	2	3	2	2			1	
20		2		2	2	2		1	
21					1				2
22	3	2		2	1	1			2
23		2		2	1	1			2
24		1		2	1	1			2
25	3			3	3	3	3		
26	4	1			1	1	4		
27	1				3				
28	3	4	3	4	4	4			4
29		1		2	1				
30	3	4				4			
31	1								
32	4	4			4		4		

Table E.2: Server dependent processing time P_{ij}

CSR/Call Type	1	2	3	4	5	6	7	8	9
1	152.4	143.4	142.8	141	187.8	177	118.8	220.8	168.6
2	189.6	147.6	142.8	144.6	183.6	186.6	173.4	220.8	168.6
3	142.8	159	142.8	119.4	174.6	201	138	220.8	168.6
4	142.8	143.4	290.4	92.4	201	177	101.4	220.8	168.6
5	142.8	207	142.8	189	237	238.8	138	220.8	233.4
6	142.8	143.4	142.8	124.8	177.6	177	138	220.8	159
7	142.8	151.8	142.8	167.4	193.8	180	138	220.8	168.6
8	160.8	143.4	142.8	147	196.8	177	126	220.8	168.6
9	142.8	143.4	142.8	141	219.6	177	138	220.8	153
10	142.8	180	142.8	152.4	204	188.4	138	220.8	258
11	156.6	143.4	142.8	141	57.6	177	129.6	220.8	168.6
12	150.6	144.6	142.8	120.6	187.8	160.8	147.6	220.8	189.6
13	142.8	143.4	142.8	126.6	188.4	165	138	220.8	288.6
14	158.4	143.4	142.8	141	87	177	114	220.8	168.6
15	153	143.4	142.8	141	187.8	177	138	220.8	168.6
16	191.4	158.4	142.8	207.6	227.4	240	147.6	220.8	168.6
17	142.8	143.4	142.8	152.4	217.8	177	138	220.8	115.8
18	173.4	120.6	142.8	172.8	181.8	173.4	138	220.8	168.6
19	161.4	126	230.4	142.8	195	177	138	273.6	168.6
20	69.6	180.6	142.8	132.6	188.4	177	138	260.4	168.6
21	142.8	143.4	142.8	141	233.4	177	138	220.8	168.6
22	114	159	142.8	143.4	153	144	84	220.8	168.6
23	142.8	184.8	142.8	186	214.8	205.8	138	220.8	210
24	147.6	111.6	142.8	128.4	168	140.4	138	220.8	168.6
25	144	142.8	142.8	141.6	186.6	175.8	153.6	220.8	168.6
26	173.4	157.8	142.8	135.6	183	176.4	187.8	25.8	168.6
27	146.4	157.8	142.8	134.4	210	177	138	220.8	168.6
28	120.6	143.4	156.6	89.4	165	177	138	210.6	168.6
29	142.8	90.6	142.8	129	153	177	138	220.8	168.6
30	142.8	143.4	142.8	141	187.8	177	138	220.8	168.6
31	234	143.4	142.8	141	187.8	177	138	220.8	168.6
32	108.6	164.4	30.6	66.6	153	165.6	104.4	220.8	168.6

APPENDIX F. ABBREVIATION LIST

ACD - Automated Call Distribution
AvgF - Average Flow Time of a Call
AvgN - Average Number of Calls
AvgW - Average Waiting Time for a call
BB - Branch and Bound
BC - Branch and Cut
BP - Branch and Price
CB - Cut and Branch
CLT - Central Limit Theorem
CMDCAW - Cutting Plane Reformulation of Problem MDCAW
CMFT - Cutting Plane Reformulation of Problem MFT
CMS - Contact Management System
CR - Continuous Relaxation
CSRs - Customer Service Representatives
CTFT - Cutting Plane Reformulation of Problem TFT
CTI - Computer Telephony Integration
DFS - Depth-First Search
FCFS - First Come First Served
FCPA - Fractional Cutting Plane Algorithm
ILP - Integer Linear Programming
INLP - Integer Nonlinear Programming
IP - Integer Programming
IVR - Interactive Voice Response
JIT - Just In Time
LB - Lower Bound
LD - Lagrangian Duality
LDMDCAW - Lagrangian Dual of Problem MDCAW
LDMFT - Lagrangian Dual of Problem MFT
LDTFT - Lagrangian Dual of Problem TFT
LP - Linear Programming
LR - Lagrangian Relaxation
Maxcldiv - Maximum Deviation of Assigned Workload
MaxF - Maximum Flow Time of a Call
MaxN - Maximum Number of calls
MaxW - Maximum Waiting Time for a Call
MDCAW - Problem of Maximum Deviation of Cumulative Assigned Workload
MED - Minimum-Expected-Delay
MFT - Problem of Maximum Flow Time

MGAP - Multiple-choice General Assignment Problem
PBX - Private Branch Exchange
RMDCAW - Reformulated Problem MDCAW
RMFT - Reformulated Problem MFT
ROMDCAW - Re-optimization of Problem MDCAW
ROMFT - Re-optimization of Problem MFT
ROTFT - Re-optimization of Problem TFT
RTFT - Reformulated Problem TFT
SBR - Skill Based Routing
SL - Service Level
SPT - Short Processing Time
TFT - Problem of Total Flow Time
UB - Upper Bound
ULS - Uncapacitated Lot Sizing
VRU - Voice Response Unit