Near-Optimal Nonmyopic Contact Center Planning Using Dual Decomposition

Akshat Kumar[†*] and **Sudhanshu Singh** and **Pranav Gupta** and **Gyana Parija**
IBM India Research Lab
[†]akshat.kumar@gmail.com, {sudsing3,prguptan,gyana.parija}@in.ibm.com
New Delhi, 110070, India

Abstract

We address the problem of minimizing staffing cost in a contact center subject to service level requirements over multiple weeks. We handle both the capacity planning and agent schedule generation aspect of this problem. Our work incorporates two unique business requirements. First, we develop techniques that can provide near-optimal staffing for 24×7 contact centers over *long term*, upto eight weeks, rather than planning myopically on a week-on-week basis. Second, our approach is usable in an online *interactive* setting in which staffing managers using our system expect high quality plans within a short time period. Results on large real world and synthetic instances show that our Lagrangian relaxation based technique can achieve a solution within 94% of optimal on an average, for eight week problems within ten minutes, whereas a generic integer programming solver can only achieve a solution within 80% of optimal. Our approach is also deployed in live business environment and reduces headcount by a decile over techniques used previously by our client business units.

1 Introduction

Cost effective staffing that meets service level requirements is crucial for profitability and sustainability of technology firms (Aksin, Armony, and Mehrotra 2007). Around of 60-80% of the operating budget of call and contact centers is spent on staffing cost (Aksin, Armony, and Mehrotra 2007; Robbins and Harrison 2010). Therefore, optimizing the staffing cost is of significant interest in such setting. Furthermore, staffing decisions are typically made weeks or months in advance to allow adequate time for hiring and training agents. This necessitates long term planning ranging from 4 to 8 weeks. The approach we develop addresses several such aspects of staffing contact centers in a real world business environment based on rigorous techniques from mathematical optimization.

Staffing in *call centers* has received significant attention within the OR and management communities. Recent literature on call center staffing and scheduling has been focussed on handling high variability in demand. The demand is assumed to be a non homogeneous Poisson process and is usually described as arrival rates over a planning horizon divided into intervals of equal size (typically ranging from 15 minutes to 1 hour). The methods require either solving a (mixed integer) LP (Bassamboo, Harrison, and Zeevi 2006) or a mix of LP and simulation. (Harrison and Zeevi 2005) solve the problem via stochastic fluid models which reduce the staffing problem to a multi-dimensional newsvendor problem and can be solved numerically by a combination of LP and Monte Carlo simulation. Stationary independent period by period (SIPP) approach finds the minimum number of servers needed to meet the service target in each interval. Schedules are then created to cover the requirements of each period. (Harrison and Robbins 2008) describe a SIPP based approach wherein they add schedules based on simulation output on top of a good initial heuristic solution. (Robbins and Harrison 2010) propose a two stage approach in which a slave problem feeds the master LP with a cutting plane to attain the optimal staffing levels. (Atlason, Epelman, and Henderson 2004) describe a two stage approach wherein simulation provides cutting planes for a master LP to solve the problem iteratively. (Ed et al. 2002) discuss a director system that integrates forecasting and planning in a single tool.

Contact centers are a generalization of call centers where a customer can send an off-line query that has to be answered within some time frame. Examples of contact centers are text and email support centers, insurance claims departments etc. The turn around time (TAT) for contact centers typically ranges from few hours to days. Hence the standard call center models which assume that contacts are answered in the interval of arrival are inappropriate for a contact center scenario. Since the peak demand can be spread over TAT, staffing requirement for a non-voice contact center (NVCC) are significantly lower than those for a call center with same incoming contact volume. Such demand spreading makes the solution space for NVCC staffing problem bigger, thus making it a difficult problem to solve.

In our work, we develop techniques for long term staffing, including capacity planning and agent scheduling, for effective contact center management. Our approach is motivated by real business constraints arising out of our engagement with clients within a large IT company with more than 20K employees. Our approach provides provably near-optimal

[*]Author presently with School of Information Systems, Singapore Management University

staffing for 24×7 contact centers over long plan horizon, upto 8-weeks. This is in contrast to approaches which plan myopically on a week-on-week basis ignoring inter-week dependencies, which can lead to violation of service level in 24×7 contact centers. Another unique business requirement is that staffing managers expect to use our tool in an *interactive* fashion. The business justification is that managers may need to tweak parameters and be able to run the planner with new parameters before deciding upon the best staffing plan. Therefore, a planner that does not provide a good solution within a few minutes is not acceptable. Our approach, based on Lagrangian relaxation, incorporates well both these requirements. It is iterative in nature providing a solution or a staffing plan approximately every minute even for large 8-week problems. The runtime and space complexity increase *linearly* with the number of weeks to plan for, resulting in a scalable approach. Furthermore, we also get a quality bound showing how far is the current solution from the optimal.

We test our approach on large real-world datasets provided by our clients as well as challenging synthetic instances. Using bounds generated by our approach, we show that our approach can achieve a solution within 94% of optimal on an average for 8-week problems within 600 seconds, whereas CPLEX can only achieve a solution within 80% of optimal within the same time limit. A variant of our approach is also deployed in the live business environment. Business validation shows that it reduces the headcount by 5-10% over custom techniques used previously by our client business units.

2 Contact Center Staffing (CSM) Problem

In this section, we describe the contact center staffing management (CSM) problem. We assume that the contact centers operate 24×7. The problem inputs are the following:

- **Granularity:** Each day is discretized into multiple time intervals of a fixed duration, say 30 minutes, also called the problem granularity G. For a scenario with 30 minute granularity, a day contains 48 consecutive time intervals indexed from 1 to 48.
- **Horizon:** Each week consists of 7 days. Horizon refers to the number of weeks the planning is done for. An 8-week problem will have $8 \times 7 \times 48 = 2688$ time intervals. Each time interval is uniquely indexed as $\langle$Week#, Day#, Interval#$\rangle$.
- **Contacts:** For every time interval, we are given the expected number of incoming call volumes or contacts. This is estimated based on historical call volume and is provided by the client.
- **TAT:** Turn-around-time or TAT refers to the business requirement that a contact should be answered within a pre-specified amount of time. The TAT is fixed for a single instance.
- **Shift Length (`sl`):** It refers to the amount of time an agent can work during the day. We can also specify lower and upper bound on the shift length.
- **Workdays (`wd`):** It refers to the number of days an agent can work in a week, usually 5.
- **AHT:** Average handling time (AHT) refers to the average time required (in minutes) to handle a contact in each of the time interval. Note that handling time for a contact depends on the interval in which it gets serviced and not on the interval of arrival. The AHT can be different for different time intervals.

Given the above inputs, the objective is to find the minimum number of agents and their schedules such that every contact arriving within problem horizon can be answered within TAT time intervals. For simplicity, in this paper we assume that an agents works 9.5 hours shift for 5 work days.

Scheduling Constraints: Agent schedules are further subject to following business constraints:

Constraint 1. *Each agent is required to work consecutive* `wd` *number of days in a week assuming weekday wraparound (explained later).*

Assuming that the week starts from Monday, weekday wraparound means that the day after Sunday is Monday of the same week. An example shift can be $\langle$Sat, Sun, Mon, Tue, Wed$\rangle$ with Thursday and Friday off of the same week.

Constraint 2. *Each day the agent logins, it works for consecutive* `sl` *time intervals.*

The concept of day is somewhat vague in 24×7 contact centers as agents can start their shifts at 11 P.M. in the night. For consistency, we consider the 24-hour interval beginning from the agent login time relevant for computing agent's shift.

Constraint 3. *An agent logins at the same time every day it is working.*

These constraints are necessary to maintain consistency in agents' weekly schedules. The schedules can change from one week to another. The above three constraints represent some of the basic business requirement. Our approach can handle their extensions as well, such as allowing agents to take any day off, login at different times of the day, allow rest times during the shift and take into account productivity factors. For ease of exposition, we have not described these additional constraints.

3 Mathematical Program for CSM

In this section, develop the outline of the mixed-integer (MIP) program for the CSM problem. The key to develop a small (w.r.t. to integer variables) MIP formulation is to efficiently handle the scheduling constraints that apply to every agent's shift. An easy way will be to create variables corresponding to a sufficiently large number of agents, assign binary variables to each agent and enforce constraints over these variables. However, this formulation will not be compact as we are assigning a unique identity to every agent in a large pool resulting in an increase in the MIP size. This fortunately is not required by realizing that every agent is identical. Therefore, we next develop novel strategies that allow us to enforce scheduling constraints compactly for each agent without requiring us to create separate sets of variables for each of them. We start with some definitions.

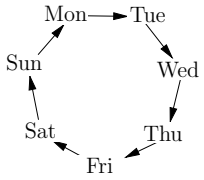


Figure 1: Circular ordering among weekdays in a week.

Definition 1. *The set* $\texttt{TI}_w$ *denotes a sorted set of all time intervals for the week* w*:*

$$\texttt{TI}_w = \{\langle w, d, i\rangle \mid d \in \{1, \ldots, 7\}, i \in \{1, \ldots, NI\}\}$$

where NI is the number of intervals in a day. For a time granularity of 30 minutes $NI = 48$. We denote the week, day and interval of $t \in \texttt{TI}_w$ by $t.w$, $t.d$ and $t.i$ respectively. Next we define an *order* function $\mathcal{O}$.

Definition 2. *Given a sequence* S*,* $\mathcal{O}(e, S)$ *returns the position of an element* e *in* S*, with positions ranging from 1 to* $|S|$*. For* $s, s' \in S$ *the notation* $s \leq s'$ *means that* $\mathcal{O}(s, S) <= \mathcal{O}(s', S)$*.*

To define the shift of an agent and address constraint 1, we introduce the notion of circular ordering among weekdays as shown in Figure 1. We define the successor of a day d as the day d' that follows the day d as per Figure 1. For example, the successor of 'Sun' is 'Mon' as per Figure 1. We denote this relation using the *successor function* $S(d) = d'$. Using the same circular ordering, we define a *distance function* $D(d, d')$ as number of directed edge traversals required to move from day d to the day d' as per Figure 1. For example, $D(\text{Sun}, \text{Mon}) = 1$ whereas $D(\text{Mon}, \text{Sun}) = 6$. The distance function is sensitive to the argument ordering and always positive. In addition, $D(d, d) = 0$ for any day d.

Definition 3. *A* 5*-day shift of an agent, starting from day* d_1*, is a sequence* $\langle d_1, d_2 = S(d_1), d_3 = S(d_2), d_4 = S(d_3), d_5 = S(d_4)\rangle$*, where* $d \in \{Mon, \ldots, Sun\}$ *and* $S(\cdot)$ *denotes the successor function.*

Using the above definition, we see that $\langle$Sat, Sun, Mon, Tue, Wed$\rangle$ is a valid shift, whereas $\langle$Sat, Mon, Tue, Thu, Fri$\rangle$ is not, as the agent does not work for 5 consecutive days. Using this notion of a shift, we define the following entities:

- Variables $x[t]\ \forall t \in \texttt{TI}_w$ denote the number of agents whose shift starts at interval $t.i$ on day $t.d$ according to the shift definition 3. To avoid double counting, we only count an agent only on the first day of its shift.
- Simple arrival window SAW$[t]\ \forall t \in \texttt{TI}_w$ denotes the set of time tuples of the last 5 days (including the current one) as per the circular ordering defined in Figure 1 with the interval number $t.i$.

$$\text{SAW}[t] = \{t' \in \texttt{TI}_w : D(t'.d, t.d) \leq 5 \ \&\ t'.i = t.i\} \quad (1)$$

- Based on SAW$[\cdot]$, we define the arrival window AW$[t]\ \forall t \in \texttt{TI}_w$ that also accounts for additional agents that are available to work in the interval t due to constraint 2:

$$\text{AW}[t] = \bigcup_{t' \in \texttt{TI}_w : 0 \leq \mathcal{O}(t, \texttt{TI}_w) - \mathcal{O}(t', \texttt{TI}_w) \leq \texttt{sl}-1} \text{SAW}[t'] \quad (2)$$

Before providing the MIP formulation, we define a few more parameters:

- $C[t]\ \forall t \in \texttt{TI}_w$ denotes the number of contacts arriving in the interval t. It is provided by the business unit.
- CHW$[t]\ \forall t \in \texttt{TI}_w$ denotes the TAT consecutive intervals starting from interval t (inclusive). Intuitively, it represents the set of available intervals in which the contacts arriving in interval t can be serviced without violating the TAT. Note that call handling window definition for last TAT-1 intervals of a week is ambiguous since the contacts arriving in those intervals can be processed next week without violating SLA. To make it unambiguous, we add TAT-1 intervals at the end of $\texttt{TI}_w$ and define CHW for last TAT-1 intervals appropriately.
- With the constructs defined above, table 1 represents the CSM problem formulation for a week. The *Meet_TAT* constraints enforce that all contacts are handled within TAT. The *Can_not_work_more_than_demand* constraints ensure that work handled upto time t does not exceed work arriving upto time t. *Can_not_work_more_than_capacity* constraints imply that minutes of work done in an interval is not more than agents minutes available.

  Due to the large TATs in NVCC setting it is acceptable, and often preferable, that some healthy backlog be maintained. Servicing all contacts arriving in a week in that week itself will require higher headcount than optimal. This implies contacts from one week flowing over to the next. Also, in 24×7 contact centers, agents working during the last $\texttt{sl} - 1$ intervals of one week may also overflow to the next week (figure 2). That is why a solution framework which solves each week's problem independently assuming that work coming in an interval is finished in that interval itself does not work for NVCC.

  To extend this problem over multiple weeks, first we define the set TI of all time intervals as

$$\text{TI} = \{\langle w, d, i\rangle | w \in \{1..N\}, d \in \{1..7\}, i \in \{1..NI\}\}$$

  where N is the number of weeks and NI is the number of intervals in a day. Replacing $\texttt{TI}_w$ by TI, the formulation in table 1 represents the full problem for several weeks. As the number of weeks increase, the multiple week problem takes exponentially higher amount of time to solve.

4 Lagrangian Relaxation for CSM-MIP

The global CSM-MIP is very hard to solve optimally due to large number of integer variables, especially for long planning horizon such as 8 weeks. Furthermore, we also show empirically that even solvers such as CPLEX can not get approximate solutions fast enough to be usable in an interactive fashion, as required in our business setting. As highlighted earlier, there are two types of inter-week dependencies, as shown in Figure 2. First, the contacts that arrive in the last TAT $-$ 1 intervals of first week can potentially overflow into the second week by remaining unserviced in the first week. This is referred to as *contact overflow* dependency. Second, according to the shift scheduling constraint 2, agents that

Variables: $W[t], x[t]$ $\quad \forall t \in \mathtt{TI}_w$

Input: $C[t], \text{AHT}[t], \text{AW}[t]$ $\quad \forall t \in \mathtt{TI}_w$

Minimize: $\sum_{t \in \mathtt{TI}_w} x[t]$

Subject to:

$Meet\_TAT$ (3)

$$\sum_{t' \in \text{CHW}[t]} W[t'] + \sum_{t' < t} W[t'] \geq \sum_{t' \leq t} C[t'] \quad \forall t \in \mathtt{TI}_w \quad (4)$$

$Can\_not\_work\_more\_than\_demand$ (5)

$$\sum_{t' \leq t} W[t'] \leq \sum_{t' \leq t} C[t'] \quad \forall t \in \mathtt{TI}_w \quad (6)$$

$Can\_not\_work\_more\_than\_capacity$ (7)

$$W[t] \cdot \text{AHT}[t] \leq G \cdot \sum_{t' \in \text{AW}[t]} x[t'] \quad \forall t \in \mathtt{TI}_w \quad (8)$$

$$W[t] \in \{0\} \cup \Re^+,\ x[t] \in \mathbb{N}_0 \quad \forall t \in \mathtt{TI}_w \quad (9)$$

Table 1: Mixed-Integer program for a single week CSM

are working in the last $\mathtt{sl} - 1$ intervals can overflow into the initial intervals of the second week. This is referred to as *agent overflow* dependency. Figure 2 shows an example with TAT$=3$ and $\mathtt{sl} = 4$.

Our strategy to obtain fast, quality bounded approximate solutions is to relax the contact and agent overflow dependencies between every two pair of adjacent weeks using the well known technique of Lagrangian relaxation (Bertsekas 1999). The resulting Lagrangian dual of the global CSM-MIP has a particularly simple structure as it is composed of multiple subproblems, one per week of the plan horizon. Thus, a dual subproblem is significantly smaller than the CSM-MIP and can be solved much faster. As the dual of the CSM-MIP provides a lower bound on the quality, we obtain quality bounds using this technique. Furthermore, we iteratively maximize the dual using its subgradient, that is essentially obtained for free by solving each dual subproblem. We refer to such a decomposition based approach as Dual Decomposition based Workforce Analytics and Planning (DD-WAP).

The DD-WAP approach satisfies a number of business constraints in our setting. First, it can handle large planning horizon as the runtime and space complexity increases *linearly* with the number of weeks, as opposed to the exponential increase in the complexity for generic MIP solvers. Furthermore, we always get quality bounds showing how far the current solution is from the optimal. Second, DD-WAP approach is iterative. Therefore, it is highly desirable in an interactive setting in which staffing managers need to repeatedly solve the problem and want to get a good solution quickly. Empirically, for 8-weeks problem, each iteration takes about one minute for real world instances.

4.1 Agent Overflow Dependency

We now describe how to relax the agent overflow dependency constraint. First, notice that only the agents that po-

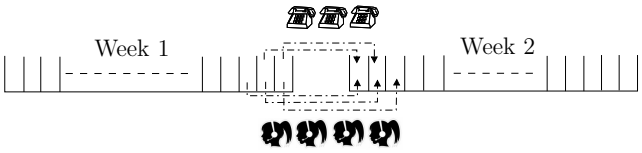


Figure 2: Inter-week dependency. Each week is divided into a number of time slots. Top arrows denote *contact overflow* dependency. Bottom arrows denote *agent overflow* dependency.

tentially work in the last $\mathtt{sl} - 1$ time intervals of a week w can potentially overflow to initial $\mathtt{sl} - 1$ time intervals of the subsequent week w'. We use the notation that subscripts w, w' index the variables corresponding to week w and w' respectively. We introduce the variables:

$$z_{w'}[t]\ \forall t \in \mathtt{TI}_{w'} : 1 \leq \mathcal{O}(t, \mathtt{TI}_{w'}) \leq \mathtt{sl} - 1$$

Intuitively, the variable $z_{w'}[t]$ denotes the number of agents who login in the last day of previous week w and are available to work in the interval t of week w'. The last $\mathtt{sl} - 1$ intervals of the week w are positioned from index $|\mathtt{TI}_w| - \mathtt{sl} + 2$ to $|\mathtt{TI}_w|$ in the sequence $\mathtt{TI}_w$. Using the base $b = |\mathtt{TI}_w| - \mathtt{sl} + 1$, we compute z variables as:

$$z_w[t'] = \sum_{t: \mathcal{O}(t, \mathtt{TI}_w) = b + \mathcal{O}(t', \mathtt{TI}_{w'})}^{|\mathtt{TI}_w|} \left(\sum_{t'' \in \text{SAW}[t]} x[t''] \right) \quad (10)$$

We create one z variable for each of the first $\mathtt{sl} - 1$ intervals of the first day of week w'. That is, for the set $\cup_{i=1}^{\mathtt{sl}-1} \{\langle w', 1, i \rangle\}$. The Eq. (10) is best understood using an example. Let us consider a 2 week problem with half hour granularity resulting in $48 \times 7 = 336$ time intervals in a single week. Also assume shift length $\mathtt{sl} = 19$ intervals (9.5 hours). The last 18 intervals of the first week are indexed from $b + 1$ to 336, where $b = 336 - 19 + 1 = 318$.

Consider the interval indexed 5 on the first day of week 2. The agents which are potentially available in this interval must have started their shift within the last $\mathtt{sl} - 1$ intervals. The earliest such interval is the interval indexed $b + 5$ of the first week. This is because an agent who logins in the interval 323 of the first week works for 14 intervals in the first week and 5 intervals in the second week, resulting in a shift length of 19. Extending this reasoning further, any agent which logins *after* interval 323 of the first week is also available to work in the interval 5 of the second week due to shift length being 19.

4.2 Demand Overflow Dependency

We now describe how to relax the demand overflow dependency constraint. First, notice that only contacts that arrive in the last TAT-1 intervals of week w can overflow to the initial TAT $- 1$ time intervals of the subsequent week w' to get serviced. We therefore introduce the variables:

$$\text{IW}_{w'}[t]\ \forall t \in \mathtt{TI}_{w'} : 1 \leq \mathcal{O}(t, \mathtt{TI}_{w'}) \leq \text{TAT} - 1$$

Intuitively, variable IW$[t]$ denotes the work that must be done in the interval t of week w' to handle unserviced contacts that overflow from week w to w'. We now develop constraints that relate variables IW and the work done in weeks

w and w'. We first introduce the notion of aggregate contacts:

$$\text{Ag}_w[k] = \sum_{t \in \texttt{TI}_w : 1 \leq \mathcal{O}(t, \texttt{TI}_w) \leq k} C_w[t] \tag{11}$$

Let TW_w denote the total work done in the week w:

$$\text{TW}_w = \sum_{t \in \texttt{TI}_w} W_w[t] \tag{12}$$

Consider the last $\text{TAT}-1$ intervals in the set $\texttt{TI}_w$. They are indexed from $b+1$ to $b+\text{TAT}-1$, where the base $b = |\texttt{TI}_w| - \text{TAT} + 1$.

Consider the interval $b+r$ for the week w. We now compute the number of contacts that have arrived until the interval $b+r$, but are not serviced in the week w:

$$\text{Ag}_w[b+r] - \text{TW}_w \tag{13}$$

Notice that the above quantity can be negative too, which implies all the contacts that have arrived until the interval $b+r$ have been handled in week w itself. The contacts which have arrived in the interval $b+r$ should be handled at most by the interval r of week w' to enforce the TAT. Therefore, we enforce the constraint that the total initial work $\text{IW}_{w'}$ until the interval r of week w' must the greater than the unserviced contacts that arrived until the internal $b+r$ of previous week w:

$$\sum_{t \in \texttt{TI}_{w'} : 1 \leq \mathcal{O}(t, \texttt{TI}_{w'}) \leq r} \text{IW}_{w'}[t] \geq \text{Ag}_w[b+r] - \text{TW}_w$$
$$\forall r = 1 \text{ to } \text{TAT}-1 \tag{14}$$

In addition to the above constraint, there are other small changes that will be described later. To summarize, we have outlined constraints that enforce agent overflow and demand overflow, namely constraints (10) and (14) respectively. Importantly, using the additional variables $z[\cdot]$ and $\text{IW}[\cdot]$, we have represented all the dependencies among two adjacent weeks in a highly separable manner. Next, we describe how to exploit such separable constraints in a Lagrangian relaxation framework.

4.3 Dualizing Dependency Constraints

First notice that for a long planning horizon problem, say 8-weeks, constraints (10) and (14) between every adjacent week summarize all the dependencies among different weeks. Therefore, our aim in this section is to show how to relax these dependency constraints in a principled manner. After relaxing these constraints, we get a dual subproblem for each week, which can be solved independently of each other leading to high scalability.

We first provide a brief overview of the Lagrangian relaxation approach, more details can be found in (Bertsekas 1999, Chapter 6). Consider the optimization problem:

$$\min f(x)$$
$$\text{s.t} \quad x \in X, \;\; g_j(x) \leq 0, \; j = 1, \ldots, r$$

The dual function $q(\cdot)$ and the Lagrangian $L(\cdot)$ of the above problem after dualizing all the constraints g are given as:

$$q(\mu) = \inf_{x \in X} L(x, \mu) = \inf_{x \in X} \big\{ f(x) + \mu \cdot g(x) \big\} \tag{15}$$

The dual solution $q(\mu)$ is a lower bound for every value of dual variables μ. The advantage while working with the dual formulation of the original problem is that the structure of the dual is often much simpler leading to computational gains. Furthermore, the dual solution also provides a lower bound on the original problem. In addition, the dual optimization problem

$$\max_{\mu : q(\mu) > -\infty} q(\mu) \tag{16}$$

is always concave and can be solved using the projected subgradient method (Bertsekas 1999). We use the notation that for a given week w, the week $\overline{w}$ denotes the week immediately before and w' denotes the week immediately after w. We use additional dummy weeks for the first and the last week of the planning horizon.

Dualizing Agent Overflow: When we dualize the constraint (10) along the lines of (15), we get the following additional term (analogous to $\mu.g(x)$) in the objective function of the weekly MIP for the week w:

$$\mathcal{A}_w(\boldsymbol{\lambda}, \boldsymbol{z}) = \sum_{r=1}^{\texttt{sl}-1} \lambda_{\overline{w}w}(r) z[t : \mathcal{O}(t, \texttt{TI}_w) = r] -$$
$$\sum_{r=1}^{\texttt{sl}-1} \lambda_{ww'}(r) \Bigg\{ \sum_{t : \mathcal{O}(t, \texttt{TI}_w) = b+r}^{|\texttt{TI}_w|} \Bigg(\sum_{t'' \in \text{SAW}[t]} x[t''] \Bigg) \Bigg\} \tag{17}$$

In the above, we use λ as the dual variables for equality constraints (10). To make the usage of subscripts clear, consider the week $w = 2$. The variable $\lambda_{ww'}$ is denoted as λ_{23}. Consider the week $w = 3$. The variable $\lambda_{\overline{w}w}$ is also λ_{23}. Therefore, using such common dual variables summarize the dependencies among different weeks. Also notice that, *all the variables* such as z and SAW in the above equation are local to the week w. For exposition purpose, we have avoided using subscripts.

Dualizing Demand Overflow: Dualizing constraint (14) leads to the following terms in the objective function of the weekly MIP for week w:

$$\mathcal{D}_w(\boldsymbol{\mu}, \mathbf{IW}, TW) = \sum_{r=1}^{\text{TAT}-1} \mu_{\overline{w}w}(r) \Bigg\{ - \sum_{t : \mathcal{O}(t, \texttt{TI}_w) = 1}^{r} \text{IW}[t] \Bigg\} +$$
$$\sum_{r=1}^{\text{TAT}-1} \mu_{ww'}(r) \text{Ag}[b+r] - TW \sum_{r=1}^{\text{TAT}-1} \mu_{ww'}(r) \tag{18}$$

where TW is the total work done during week w. Again, all the variables are local to the week w. Using the quantities $\mathcal{A}$ and $\mathcal{D}$, evaluating the dual function $q(\boldsymbol{\lambda}, \boldsymbol{\mu})$ becomes much easier and reduces to solving a single MIP for each week w:

$$q(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \sum_{w=1}^{\#Weeks} q_w(\boldsymbol{\lambda}_w, \boldsymbol{\mu}_w) \tag{19}$$

where $q_w(\cdot)$ is the solution of the optimization problem shown in Table 2. Such a decomposable structure of the dual function is the key to significant gain in scalability

Variables: $W[\cdot], x[\cdot], z[\cdot], \text{IW}[\cdot], \text{IWT}, TW$

Input: $C[\cdot], \text{AHT}[\cdot], \text{AW}[\cdot], \boldsymbol{\lambda}, \boldsymbol{\mu}$

Minimize: $\sum_{t \in \texttt{TI}_w} x[t] + \mathcal{A}_w(\boldsymbol{\lambda}, \boldsymbol{z}) + \mathcal{D}_w(\boldsymbol{\mu}, \mathbf{IW}, TW)$

Subject to:

Meet_TAT:

$$\sum_{t' \in \text{CHW}[t]} W[t'] + \sum_{t'<t} W[t'] \geq \sum_{t' \leq t} C[t'] + \text{IWT} \; \forall t \in \texttt{TI}_w$$

Meet_InitWorkLoad:

$$\sum_{t' \leq t} W[t'] \geq \sum_{t' \leq t} \text{IW}[t'] \; \forall t : \mathcal{O}(t, \texttt{TI}_w) = 1, \ldots, \text{TAT}-1$$

Cannot_Work_More_Than_Demand:

$$\sum_{t' \leq t} W[t'] \leq \sum_{t' \leq t} C[t'] + \text{IWT} \qquad \forall t \in \texttt{TI}_w$$

Cannot_Work_More_Than_Capacity1:

$$W[t] \cdot \text{AHT}[t] \leq G \cdot \left\{ \sum_{t' \in \text{AW}[t]} x[t'] + z[t] \right\} \forall t : \mathcal{O}(t, \texttt{TI}_w) \leq \texttt{sl} - 1$$

Cannot_Work_More_Than_Capacity2:

$$W[t] \cdot \text{AHT}[t] \leq G \cdot \sum_{t' \in \text{AW}[t]} x[t'] \forall t \in \texttt{TI}_w \; : \mathcal{O}(t, \texttt{TI}_w) \geq \texttt{sl}$$

Total_Init_Work_Load

$$\text{IWT} = \sum_{t : \mathcal{O}(t, \texttt{TI}_w) = 1}^{\text{TAT-1}} \text{IW}[t]$$

Total_Work_Done:

$$TW = \sum_{t \in TI} W[t]$$

$$W[t] \in \{0\} \cup \Re^+, \; x[t] \in \mathbb{N}_0 \forall t \in \texttt{TI}_w$$

$$z[t] \in \mathbb{N}_0 \forall t : \mathcal{O}(t, \texttt{TI}_w) \leq \texttt{sl} - 1, \text{IW}[\cdot] \in \{0\} \cup \Re^+$$

Table 2: MIP for solving the dual subproblem for week w

for large planning horizon over the global MIP formulation. Constraints *Meet_TAT* in Table 2 enforce that TAT is met for contacts arriving in the current week and for the entire backlog coming from previous week (IWT). *Meet_InitWorkLoad* ensure correct computation of workload overflowing from previous week. *Cannot_Work_More_Than_Demand* imply that work done upto any time interval can not exceed work arriving upto that interval. The *Cannot_Work_More_Than_Capacity1* constraints for first $\texttt{sl} - 1$ intervals state that minutes of work done is limited by agents coming over from previous week and agents logging in the current week. *Cannot_Work_More_Than_Capacity2* state that after the first $\texttt{sl} - 1$ intervals, minutes of work done in an interval is limited by agents logging in the current week only. Remaining constraints are self explanatory. Note that for the decomposed dual problem time intervals are defined over a week only, with TAT-1 extended intervals added at the end of week.

4.4 Subgradient Based Dual Optimization

We now address the problem of optimizing the dual:

$$\max_{\boldsymbol{\lambda}, \boldsymbol{\mu} : q(\boldsymbol{\lambda}, \boldsymbol{\mu}) > -\infty} q(\boldsymbol{\lambda}, \boldsymbol{\mu}) \tag{20}$$

We maximize the dual iteratively by using the projected sub gradient technique (Bertsekas 1999). The variables λ, for each $r = 1, \ldots, \texttt{sl} - 1$, are updated after each iteration as:

$$\lambda^{\star}_{w,w+1}(r) \leftarrow \lambda_{w,w+1}(r) + \alpha \Bigg[\overline{z}_{w+1}[t' : \mathcal{O}(t', \texttt{TI}_{w'}) = r] - \sum_{t : \mathcal{O}(t, \texttt{TI}_w) = b + r}^{|\texttt{TI}_w|} \left(\sum_{t' \in \text{SAW}_w[t]} \overline{x}_w[t'] \right) \Bigg] \; \forall w = 1 \ldots H - 1 \tag{21}$$

where H denotes the problem horizon, and $\overline{z}_{w+1}$ denote the solution of MIP in Table 2 for the week $w+1$ and $\overline{x}_w$ denote the solution for the week w and α is the step parameter. The parameters μ, for each $r = 1, \ldots, \text{TAT}-1$, are updated as:

$$\mu^{\star}_{w,w+1}(r) \leftarrow \left(\mu_{w,w+1}(r) + \alpha \Bigg[\text{Ag}_w[b + r] - \overline{\text{TW}}_w - \sum_{t \in \text{TI}_{w+1} : \mathcal{O}(t, \text{TI}_{w+1}) = 1}^{r} \overline{\text{IW}}_{w+1}[t] \Bigg] \right)_+ \forall w = 1 \ldots H - 1 \tag{22}$$

where $()_+$ denotes the projection on the positive (≥ 0) real line, variables with overhead bar denote the solution obtained from the corresponding MIP in Table 2. The step parameter α is calculated using strategies highlighted in (Bertsekas 1999, Section 6.3.1).

As the dual solution obtained may not be a feasible primal solution, we use a number of strategies (omitted due to space constraints) to extract a consistent primal solution from the dual. These strategies involve solving a series of small MIPs, one for each week. This step, therefore, is also scalable with the plan horizon.

5 Experiments

We test our DD-WAP approach on multiple real world as well as synthetic datasets. All the instances were 8-week problem with a granularity of 30 minutes. We compare our solver with CPLEX, which was used to solve the global MIP formulation. Each dual subproblem in our approach was also solved using CPLEX. All our experiments were performed on a dual core PC with 8GB RAM and 2.6GHz CPU. The DD-WAP solver was implemented in JAVA.

5.1 Real World Datasets

We had access to two real world datasets provided by our client business units. The first dataset comprises the actual demand and handling time information for each time interval for 30 consecutive weeks. The second datset had the same information for 12 weeks. We created 20 different 8-week instances from the first dataset and 10 different 8-week instances from the second dataset. The shift length was 19 intervals, implying a 9.5 hour shift for each agent and the TAT

CPLEX		DD-WAP		
Primal	Dual	Primal	Dual	% Gain
460	339.50	368	343.10	20.00%
517	394.20	423	389.92	18.18%
404	390.50	413	389.91	-2.23 %
389	374.25	412	373.25	-5.91 %
466	366.86	395	371.74	15.24%
364	354.71	382	357.00	-4.95 %
475	362.94	386	367.51	18.74%
487	358.12	373	361.64	23.41%
461	356.86	388	356.34	15.84%
434	359.25	403	363.05	7.14%
460	339.50	365	343.07	20.65%
-	184.95	200	188.90	100.00%
517	394.20	414	389.92	19.92%
404	390.50	415	389.91	-2.72 %
389	374.25	409	373.25	-5.14 %
466	366.87	393	371.74	15.67%
417	354.71	380	357.00	8.87 %
475	362.95	383	367.51	19.37%
487	358.13	379	361.64	22.18%
461	356.87	385	356.34	16.49%

Table 3: Solution quality comparisons between CPLEX and DD-WAP for *real dataset* 1 after 10 minutes, total 20 instances. A '-' indicates no solution was found within the time limit.

was 10 intervals or 5 hours. Each agent was required to work 5 days per week. Our clients had a strict time limit of 5 minutes for getting the solution. This was required so as to use the planner in an interactive fashion. For the purpose of this work, we use a deadline of 10 minutes. Our decomposition based approach has enough scope for parallelism as each dual subproblem can be solved in parallel. Therefore, an improved *parallel* version of our current sequential approach can meet the 5 minute deadline.

Table 3 shows the solution quality comparisons for CPLEX and the DD-WAP approach. We report the best primal and dual achieved by each approach within 10 minutes. The primal solution quality denotes the total number of agents required for the 8-weeks. Therefore, a lower primal value is better. Similarly, a higher dual value is better as the dual is a lower bound on the primal. The last column shows the gain in solution quality $((\text{Primal}_{\text{CPLEX}} - \text{Primal}_{\text{DD\_WAP}})/\text{Primal}_{\text{CPLEX}})$. A positive gain value indicates that the DD-WAP approach is better. This gain value shows that DD-WAP approach consistently gives significantly better results than the CPLEX for most instances.

Figure 3 shows the best solution quality achieved by DD-WAP as a percentage of the optimal, computed using the bounds provided by the DD-WAP approach. This figure clearly shows that the DD-WAP approach provides provably near-optimal solution (about 94% of optimal) for all the instances. This is encouraging news for our approach which is much more scalable w.r.t. the planning horizon than the global MIP formulation solved using CPLEX. The CPLEX was only able to provide a solution within 80% of the optimal on average for these instances within 10 minutes. This clearly shows the ability of our approach to get a good solution significantly faster than CPLEX.

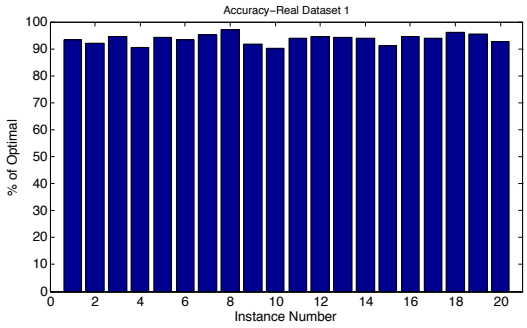


Figure 3: Solution quality as a percentage of optimal for the DD-WAP approach after 10 minutes for *real dataset 1*

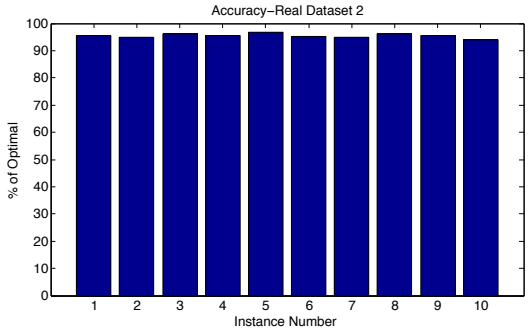


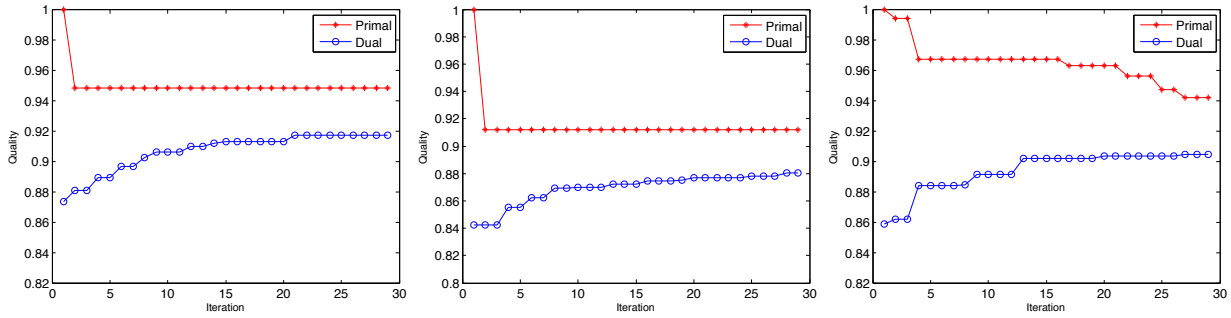
Figure 4: Solution quality as a percentage of optimal for the DD-WAP approach after 10 minutes for *real dataset 2*

Table 4 shows the solution quality comparisons for the *real dataset* 2. This table clearly shows the advantage of our approach over CPLEX. The CPLEX is unable to provide a feasible solution with the time limit of 10 minutes for many instances. The column 'When Feasible' shows the time (in seconds) when CPLEX first provided a feasible solution for the global MIP. This column shows that CPLEX takes significantly more time than the allowed 600 seconds limit to even get a feasible solution. This further validates that our approach is going to be more useful in a time limited interactive setting.

Figure 4 shows the best solution quality achieved by

CPLEX			DD-WAP	
Primal	Dual	When Feasible	Primal	Dual
-	-	839.00	765	731.63
-	-	1111.46	787	748.00
869	774.33	516.00	802	770.58
-	-	932.45	804	768.34
-	-	714.84	785	758.56
755	739.49	483.18	769	731.50
-	-	712.10	744	706.49
-	-	942.90	727	699.28
-	-	972.81	714	681.00
-	-	764.92	711	669.00

Table 4: Solution quality comparisons between CPLEX and DD-WAP for *real dataset* 2 after 10 minutes, total 10 instances. A '-' indicates no feasible solution was found within the time limit.



(a) Average demand 65 (b) Average demand 75 (c) Average demand 85

Figure 5: Best Primal and Dual solution quality (synthetic instances) with iteration. Each value is scaled to be ≤ 1.

Synthetic Instances		
Mean	10 Min. Gap	Final Gap
65	3.78%	2.40%
75	5.14%	4.14%
85	4.89%	2.70%

Table 5: Solution quality results on synthetic instances for DD-WAP. 'Mean' denotes the average contacts per time interval. 'Gap' denotes the distance from the optimal solution.

DD-WAP as a percentage of the optimal, computed using the bounds provided by the DD-WAP approach. This figure shows that DD-WAP approach provides provably near-optimal solution (within 95% of optimal) for all the 10 instances. This result further illustrates the ability of our approach to get a good solution significantly faster than CPLEX. Furthermore, we have also deployed a simplified version of the DD-WAP solver with our client business units, which provides 5-10% headcount reduction over their previous methods.

5.2 Synthetic Instances

We generated a number of synthetic 8-week benchmarks based on properties of the real world instances. The average number of contacts arriving per time interval for real world instances was 70. Therefore, we vary the average contact parameter from 65 to 85. We use the same handling time as in the real world instances. We sample the number of contacts arriving per interval from the gamma distribution, which is often used to model queueing systems. We use the shape parameter k of this gamma distribution as 3 to allow enough variations in the number of contacts across different time intervals. The scale parameter θ was chosen to be 'Mean'$/k$, where 'mean' is the required average number of contacts (65, 75 or 85). The expected value of a gamma distribution is $k\theta$, resulting in the required average value.

Table 5 shows the solution quality results for DD-WAP for varying average demand (65, 75 or 85). We generated 5 instances for each average demand setting, resulting in a total of 15 8-week instances. The '10 min. gap' column shows the average gap (over 5 instances) between the best DD-WAP solution and the optimal computed using the bounds provided by DD-WAP. The'Final Gap' column shows the final gap after 29 iterations of our solver. This table further demonstrates that our approach can quickly provide near-optimal solution for a range of settings. The CPLEX was unable to provide a feasible solution within 10 minutes time limit and is therefore not reported.

Figure 5 shows how the primal and dual solution quality evolve with iterations of DD-WAP for three different instances, one for each mean demand of 65, 75 and 85. These instances were chosen to highlight how the large initial gap between the primal and the dual solution in the first iteration, about 14% for each of the three instances, can be effectively reduced by the DD-WAP approach to less than 4% at the end of 29 iterations. These results further illustrate the utility of the DD-WAP approach even for difficult instances where the initial gap between the primal and the dual solution is high.

6 Conclusion and Future Directions

In this work, we addressed the contact center staffing problem. Our approach, based on Lagrangian relaxation, addressed the two key business constraints. First, it is highly scalable with the length of the planning horizon and is able to find provably good quality solutions for such large planning problems. Second, our approach is applicable in an interactive setting when a good solution is desired quickly by staffing managers. Our approach provided near-optimal solutions within 10 minutes for complex real life data instances, while CPLEX solving the global MIP struggled to even find a feasible integer solution within 10 minutes. The DD-WAP approach is scalable since the runtime and space complexity increases linearly with the number of weeks. Our key contributions are thus formulating the problem of contact center management (CSM) in a MIP framework and providing an efficient iterative approach based on dual decomposition to solve the CSM problem.

In this paper we have considered a simpler version of the CSM problem where in agents work consecutive days, log in at the same time every day. Real business settings require capabilities to allow flexible login at multiple times as well flexible work days. With all these constraints added, the full problem formulation can take up to a day to get a solution within 90% of the optimal. As more such business constraints are added, the scalability and anytime nature of the DD-WAP approach would offer even greater advantage.

References

Aksin, O.; Armony, M.; and Mehrotra, V. 2007. The modern call-center: A multi-disciplinary perspective on operations management research. *Production and Operations Management* 16:665–688.

Atlason, J.; Epelman, M. A.; and Henderson, S. G. 2004. Call center staffing with simulation and cutting plane methods. *Annals of Operations Research* 127:333 – 358.

Bassamboo, A.; Harrison, J.; and Zeevi, A. 2006. Design and control of a large call center: Asymptotic analysis of an lp-based method. *Operational Research* 54(3):419 – 435.

Bertsekas, D. P. 1999. *Nonlinear Programming*. Athena Scientific, 2nd edition.

Ed, A. F.; Fama, J.; Andre, D.; Matan, O.; and Nourbakhsh, I. 2002. Staff scheduling for inbound call centers and customer contact centers. In *In Eighteenth National Conference on Artificial intelligence*, 822–829.

Harrison, T., and Robbins, T. 2008. A simulation based scheduling model for call centers with uncertain arrival rates. In *Proceedings of the 2008 Winter Simulation Conference*, 2884 – 2890.

Harrison, J., and Zeevi, A. 2005. A method for staffing large call centers based on stochastic fluid models. *Manufacturing and Service Operations Management* 7:20 – 36.

Robbins, T. R., and Harrison, T. P. 2010. A stochastic programming model for scheduling call centers with global service level agreements. *European Journal of Operational Research* 207(3):1608–1619.