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 Mehrrota 2007). Around of 60-80% of the operating budget of call and contact centers is spent on staffing cost (Aksin, Armony, and Mehrrota 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 focused 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
staffing for 24 × 7 contact centers over long plan horizon, up to 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 × 7 × 48 = 2688 time intervals. Each time interval is uniquely indexed as ⟨Week#, Day#, Interval#⟩.
- **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 agent 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 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 ⟨Sat, Sun, Mon, Tue, Wed⟩ with Thursday and Friday off of the same week.

**Constraint 2.** Each day the agent logs in, it works for consecutive 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 logs in 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.
Definition 1. The set $\mathcal{T}_{I_w}$ denotes a sorted set of all time intervals for the week $w$:

$$\mathcal{T}_{I_w} = \{ (w, d, i) | d \in \{1, \ldots, 7\}, i \in \{1, \ldots, NI\} \}$$

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 $t \in \mathcal{T}_{I_w}$ 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 $\mathcal{O}(s, S) \leq \mathcal{O}(s', S)$.

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, $D(Sun, Mon) = 1$ whereas $D(Mon, Sun) = 6$. 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 (Sat, Sun, Mon, Tue, Wed) is a valid shift, whereas (Sat, Mon, Tue, Thu, Fri) is not, as the agent does not work for 5 consecutive days. Using this notion of a shift, we define the following entities:

- Variables $x[t] \forall t \in \mathcal{T}_{I_w}$ 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 $SAW[t] \forall t \in \mathcal{T}_{I_w}$ 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 $t.i$.

$$SAW[t] = \{ t' \in \mathcal{T}_{I_w} | D(t', d, t.d) \leq 5 \& t'.i = t.i \}$$  \hspace{1cm} (1)

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

$$AW[t] = \bigcup_{t' \in \mathcal{T}_{I_w} : 0 \leq \mathcal{O}(t, t.i) - \mathcal{O}(t', t.i) \leq s\leq 1} SAW[t']$$  \hspace{1cm} (2)

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

- $C[t] \forall t \in \mathcal{T}_{I_w}$ denotes the number of contacts arriving in the interval $t$. It is provided by the business unit.

- $CHW[t] \forall t \in \mathcal{T}_{I_w}$ 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 $\mathcal{T}_{I_w}$ 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 up to time $t$ does not exceed work arriving up to 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 $24 \times 7$ contact centers, agents working during the last $s_1 - 1$ 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

$$TI = \{ (w, d, i) | w \in \{1..N\}, d \in \{1..7\}, i \in \{1..NI\} \}$$

where $N$ is the number of weeks and $NI$ is the number of intervals in a day. Replacing $\mathcal{T}_{I_w}$ 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
We now describe how to relax the demand overflow dependency. First, notice that only the agents that po-
tentially work in the last \( s_1 - 1 \) time intervals of a week \( w \) can potentially overflow to initial \( s_1 - 1 \) 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:

\[
z_w[t] \quad \forall t \in T(\cdot, T(w)) : 1 \leq O(t, TI_w) \leq s_1 - 1
\]

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 \( s_1 - 1 \) intervals of the week \( w \) are positioned from index \([TI_w] - s_1 + 2 \) to \([TI_w] - 1\) in the sequence \( TI_w \). Using the base \( b = [TI_w] - s_1 + 1 \), we compute \( z \) variables as:

\[
z_w[t'] = \sum_{t:O(t,TI_w)=b,O(t',TI_w')} \sum_{t'' \in SAW[t]} x[t'']
\]

We create one \( z \) variable for each of the first \( s_1 - 1 \) intervals of the first day of week \( w' \). That is, for the set \( \bigcup_{i=1}^{s_1-1} \{ (w', 1, i) \} \). 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 \( s_1 = 19 \) 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 \( s_1 - 1 \) intervals. The earliest such interval is the interval indexed \( b + 5 \) of the first week. This is because an agent who logs 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 logs in 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:

\[
IW_w[t] \quad \forall t \in T(\cdot, T(w)) : 1 \leq O(t, TI_w) \leq TAT - 1
\]

Intuitively, variable \( IW_w[t] \) 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
and $w'$. We first introduce the notion of aggregate contacts:

$$Ag_w[k] = \sum_{t \in \mathcal{T}_w \mid 1 \leq \mathcal{O}(t, \mathcal{T}_w) \leq k} C_w[t]$$

(11)

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

$$TW_w = \sum_{t \in \mathcal{T}_w} W_w[t]$$

(12)

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

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 $w$:

$$Ag_w[b + r] - TW_w$$

(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 $IW'_w$ until the interval $r$ of week $w'$ must be the greater than the unserviced contacts that arrived until the interval $b + r$ of previous week $w$:

$$\sum_{t \in \mathcal{T}_w \mid 1 \leq \mathcal{O}(t, \mathcal{T}_w) \leq r} IW_w[t] \geq Ag_w[b + r] - TW_w \quad \forall r = 1 \text{ to } TAT - 1$$

(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 $z[.]$ and $IW[.]$, 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:

$$\min f(x)$$

s.t $x \in X$, $g_j(x) \leq 0$, $j = 1, \ldots, r$

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

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

(15)

The dual solution $q(\mu)$ is a lower bound for every value of dual variables $\mu$. 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

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

(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 $\mathcal{W}$ 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 $\mu \cdot g(x)$) in the objective function of the weekly MIP for the week $w$:

$$A_w(\lambda, z) = \sum_{r=1}^{TAT-1} \lambda{\pi}_{w}(r)z[t : \mathcal{O}(t, \mathcal{T}_w) = r] -$$

$$\sum_{r=1}^{TAT-1} \lambda{\pi}_{w'}(r) \left\{ \sum_{t : \mathcal{O}(t, \mathcal{T}_w) = b + r} \sum_{t' : \mathcal{S}\mathcal{A}\mathcal{W}[t]} x[t'] \right\}$$

(17)

In the above, we use $\lambda$ as the dual variables for equality constraints (10). To make the usage of subscripts clear, consider the week $w = 2$. The variable $\lambda_{w'}$ is denoted as $\lambda_{23}$. Consider the week $w = 3$. The variable $\lambda_{w'}$ is also $\lambda_{23}$. 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 $w$:

$$D_w(\mu, IW, TW) = \sum_{r=1}^{TAT-1} \mu{\pi}_{w}(r) \left\{ \sum_{t : \mathcal{O}(t, \mathcal{T}_w) = 1} IW[t] \right\} +$$

$$\sum_{r=1}^{TAT-1} \mu{\pi}_{w'}(r) Ag[b + r] - TW \sum_{r=1}^{TAT-1} \mu{\pi}_{w'}(r)$$

(18)

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

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

(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: \( W_t, x_t, z_t \), \( IW_t \), \( IWT \), \( TW \)
Input: \( C_t, AHT_t, AW_t \), \( \lambda, \mu \)
Minimize: \( \sum_{t \in IT_{\lambda}} x_t + A_w(\lambda, z) + D_w(\mu, IW, TW) \)
Subject to:
- \( \sum_{t' \in CH_t} W_{t'} + \sum_{t' \leq t} W_{t'} \geq \sum_{t' \leq t} C_{t'} + IWT \forall t \in IT_{\lambda} \)
- \( \sum_{t' \leq t} W_{t'} \geq \sum_{t' \leq t} IW_{t'} \forall t : O(t, IT_{\lambda}) = 1, \ldots, TAT - 1 \)
- \( \sum_{t' \leq t} W_{t'} \leq \sum_{t' \leq t} C_{t'} + IWT \forall t \in IT_{\lambda} \)
- \( W[t] \cdot AHT[t] \leq G \cdot \left( \sum_{t' \in AW[t]} x[t'] + z[t] \right) \forall t : O(t, IT_{\lambda}) \leq s1 - 1 \)
- \( W[t] \cdot AHT[t] \leq G \cdot \left( \sum_{t' \in AW[t]} x[t'] \right) \forall t \in IT_{\lambda} : O(t, IT_{\lambda}) \geq s1 \)

\( IWT = \sum_{t : O(t, IT_{\lambda}) = 1} IW[t] \)

\( TW = \sum_{t \in T} W[t] \)

\( W[t] \in \{0\} \cup \mathbb{R}^+ \), \( x[t] \in \mathbb{N}_0 \forall t \in IT_{\lambda} \)

\( z[t] \in \mathbb{N}_0 \forall t : O(t, IT_{\lambda}) \leq s1 - 1, IW[t] \in \{0\} \cup \mathbb{R}^+ \)

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 \( s1 - 1 \) intervals state that minutes of work done is limited by agents coming over from previous week and agents logging in the current week. \( Cannot, Work, More, Than, Capacity1 \) state that after the first \( s1 - 1 \) 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:

\[
\max_{\lambda, \mu} q(\lambda, \mu) > -\infty
\]

We maximize the dual iteratively by using the projected subgradient technique (Bertsekas 1999). The variables \( \lambda \) for each \( r = 1, \ldots, s1 - 1 \), are updated after each iteration as:

\[
\lambda_{w,w+1}(r) \leftarrow \lambda_{w,w+1}(r) + \alpha \left[ O_{w+1}(t', IT_{w+1}) = r \right] - \sum_{t : O(t, IT_{w+1}) = b + r} \left( \sum_{t' \in SAW[t]} \pi_{w,t'} \right) \forall w = 1 \ldots H - 1
\]

where \( H \) denotes the problem horizon, and \( \pi_{w+1} \) denote the solution of MIP in Table 2 for the week \( w + 1 \) and \( \pi_{w} \) denote the solution for the week \( w \) and \( \alpha \) is the step parameter. The parameters \( \mu_t \) for each \( r = 1, \ldots, TAT - 1 \), are updated as:

\[
\mu_{w,w+1}(r) \leftarrow \mu_{w,w+1}(r) + \alpha \left[ A_{w,b} + r - TW_w = 1 \right] + \sum_{t : O(t, IT_{w+1}) = 1} \pi_{w+1,t} \right] \forall w = 1 \ldots H - 1
\]

where \( r \) denotes the projection on the positive \( (\geq 0) \) real line, variables with overhead bar denote the solution obtained from the corresponding MIP in Table 2. The step parameter \( \alpha \) 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 dataset 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.
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.

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

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 ((Primal_{CPLEX} − Primal_{DD-WAP})/Primal_{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.

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 and DD-WAP for real dataset 2 after 10 minutes, total 20 instances. A '-' indicates no feasible solution was found within the time limit.

<table>
<thead>
<tr>
<th>Instance Number</th>
<th>Primal</th>
<th>Dual</th>
<th>When Feasible</th>
<th>Primal</th>
<th>Dual</th>
</tr>
</thead>
<tbody>
<tr>
<td>460</td>
<td>339.50</td>
<td>368</td>
<td>839.00</td>
<td>765</td>
<td>731.63</td>
</tr>
<tr>
<td>517</td>
<td>394.20</td>
<td>423</td>
<td>1111.46</td>
<td>787</td>
<td>748.00</td>
</tr>
<tr>
<td>404</td>
<td>390.50</td>
<td>413</td>
<td>516.00</td>
<td>802</td>
<td>770.58</td>
</tr>
<tr>
<td>389</td>
<td>374.25</td>
<td>412</td>
<td>932.45</td>
<td>804</td>
<td>768.34</td>
</tr>
<tr>
<td>466</td>
<td>366.86</td>
<td>395</td>
<td>714.84</td>
<td>785</td>
<td>758.56</td>
</tr>
<tr>
<td>364</td>
<td>354.71</td>
<td>382</td>
<td>714.20</td>
<td>744</td>
<td>706.49</td>
</tr>
<tr>
<td>475</td>
<td>362.94</td>
<td>386</td>
<td>942.90</td>
<td>727</td>
<td>699.28</td>
</tr>
<tr>
<td>487</td>
<td>358.12</td>
<td>373</td>
<td>972.81</td>
<td>714</td>
<td>681.00</td>
</tr>
<tr>
<td>461</td>
<td>356.86</td>
<td>388</td>
<td>764.92</td>
<td>711</td>
<td>669.00</td>
</tr>
</tbody>
</table>

Table 4: Solution quality comparisons between CPLEX and DD-WAP for real dataset 2 after 10 minutes, total 20 instances. A '-' indicates no feasible solution was found within the time limit.
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.

<table>
<thead>
<tr>
<th>Synthetic Instances</th>
<th>Mean 10 Min. Gap</th>
<th>Final Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>65</td>
<td>3.78%</td>
<td>2.40%</td>
</tr>
<tr>
<td>75</td>
<td>5.14%</td>
<td>4.14%</td>
</tr>
<tr>
<td>85</td>
<td>4.89%</td>
<td>2.70%</td>
</tr>
</tbody>
</table>

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 \( \theta \) 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


