Contact Center Scheduling with Strict Resource Requirements

Aman Dhesi[1], Pranav Gupta[2], Amit Kumar[3], Gyana R. Parija[2], and Sambuddha Roy[2]

[1] Department of Computer Science, Princeton University
adhesi@princeton.edu
[2] IBM Research – India, New Delhi
{prguptan,gyana.parija,sambuddha}@in.ibm.com
[3] Department of Computer Science and Engg., Indian Institute of Technology, Delhi
amitk@cse.iitd.ac.in

Abstract. Consider the following problem which often arises in contact center scheduling scenarios. We are given a set of employees where each employee can be deployed for shifts consisting of L consecutive time units. Further, each employee specifies a set of possible start times, and can be deployed for a bounded number of shifts only. At each point of time t, we are also given a lower bound r_t on the number of employees that should be present at this time. The goal is to find a schedule for the employees such that the number of time slots whose requirements are met is maximized. Such problems naturally arise in many other situations, e.g., sensor networks and cloud computing.

The strict nature of the resource requirement makes this problem very hard to approximate. In this paper, we give a bicriteria approximation algorithm for this problem. Given a parameter $\varepsilon > 0$, we give an $O(\frac{1}{\varepsilon^3} \cdot \log \frac{1}{\varepsilon})$-approximation algorithm for this problem, where we count those time slots for which we satisfy at least $(1-\varepsilon)$-fraction of the requirement. Our techniques involve a configuration LP relaxation for this problem, and we use non-trivial structural properties of an optimal solution to solve this LP relaxation. We even consider the more general problem where shift lengths of different employees can vary significantly. In this case, we show that even finding a good bicriteria approximation is hard (under standard complexity theoretic assumptions).

1 Introduction

Scheduling employees is a central problem in workforce management in contact centers [ICWC10,FHF+02]. Typically, in these settings, we have a reasonably accurate forecast of demands which will arrive at any point of time (also referred as *time slots*) in future. This in turn implies that we roughly know, for each time slot, how many employees will be needed. The employees are usually required to work for shifts of a certain fixed duration, which we shall denote by L. Many of these employees work as part-time workers and often have varying constraints on their availability. In other words, for each employee, we know at what times they

can possibly start on a shift. Given these constraints, we would like to schedule these employees such that we maximize the number of timeslots for which we have the requisite number of employees. The setting of contact center often demands, via service level agreements (SLA's), that we satisfy the requirements of a large fraction of timeslots.

The framework that we consider in this paper is quite general and can capture many other situations :

- Sensor networks : Sensors have limited power and can last for only a certain duration. In many cases, we would like to use a sensor non-preemptively for its entire lifetime. Suppose we are given, for each time unit, how many sensors we would like to be active at that time. The problem is now to schedule the sensors (i.e., decide when to turn them on) such that we maximize the number of timeslots whose requirements are met.
- Cloud computing : In cloud computing, unit size tasks arrive over a period of time, and we process these by assigning them to a set of processors available over a network. Various constraints over availability of a processor may dictate that we can only use them in one (or several) possible slots for a certain period of time.
- Energy Management: We are given the energy requirement for each point of time in future, and are also given a set of available energy sources. However, each energy source can run for only a fixed duration of time, and may be available at some specific timeslots only.

We now define the problem more concretely. As mentioned earlier, we divide time into slots : let T denote the set of time-slots. We shall often refer the employees as *resources* – let R denote the set of resources. For each time $t \in T$, we are given a requirement r_t, and a profit p_t. For each resource $i \in R$, we are given a parameter α_i and a set of intervals $\mathcal{E}_i$, where each interval in $\mathcal{E}_i$ is of uniform length L. We would like to select at most α_i intervals from $\mathcal{E}_i$ for each i such that the number of *satisfied* timeslots is maximized. A timeslot is satisfied if there are at least r_t selected intervals, no two from the same set $\mathcal{E}_i$, which contain t. We call this problem the `MaxStaff` problem.

Related Work There has been much work on workforce management in the operations research community [AAM07,GKM03]. Contact center scheduling presents new kinds of constraints where employee availability (and skills) have very high variance [Rey03]. To the best of our knowledge, ours is the first work to view these problems from an approximation algorithms perspective.

The `MaxStaff` problem is a mixed packing-covering problem – the constraints on resources are of packing type, whereas we need to cover the demands. If we relax the strict nature of the requirements, i.e., if we are just interested in maximizing the sum over all timeslots of the *fraction* to which it is satisfied, then the problem becomes a packing problem. In fact, in this case, it falls under the framework of maximizing a sub-modular function under matroid and knapsack constraints [CCPV07,GRST10,LMNS09].

Our Contributions Our first result is that the `MaxStaff` problem is as hard to approximate as the `Maximum Independent Set` problem.

Theorem 1. *There is an approximation preserving reduction from the* `Maximum Independent Set` *problem to the* `MaxStaff` *problem.*

Since very strong hardness results are known for the `Maximum Independent Set` problem [ST00], we look for bicriteria approximation algorithms of the following kind. Given an instance $\mathcal{I}$ of the `MaxStaff` problem, let $\texttt{opt}(\mathcal{I})$ denote the cost of the optimum solution for $\mathcal{I}$. For a parameter γ, $0 \leq \gamma \leq 1$, we say that a solution γ-satisfies a timeslot t if there are at least $\gamma \cdot r_t$ selected intervals containing t, no two from the same set $\mathcal{E}_i$ for any i. Given parameters α, γ, $0 \leq \gamma \leq 1$, we say that a solution is (α, γ)-bicriteria approximate if the total profit of timeslots which are γ-satisfied by this solution is at least $\alpha \cdot \texttt{opt}(\mathcal{I})$. An (α, γ)-bicriteria algorithm is one which produces solutions with this property. We show that for any constant ε, we can get a $(poly(\varepsilon), 1-\varepsilon)$-bicriteria approximation algorithm for this problem.

Theorem 2. *For any constant $\varepsilon > 0$, there is a poly-time $\left(O\left(\frac{\varepsilon^3}{\log \frac{1}{\varepsilon}}\right), 1-\varepsilon\right)$-bicriteria approximation algorithm for the* `MaxStaff` *problem.*

We then consider the case when the shift length L may vary over different elements in R, i.e., if the intervals in $\cup_{i \in R} \mathcal{E}_i$ can be of arbitrary lengths. On the positive side, the proof of Theorem 2 can be easily extended to show that if the ratio of the maximum length to the minimum length of an interval is β, then one can get an $\left(O\left(\frac{\varepsilon^3 \cdot \beta}{\log \frac{1}{\varepsilon}}\right), 1-\varepsilon\right)$-bicriteria approximation algorithm. However, we show that in this case, one cannot hope for a significantly better result.

Theorem 3. *For any small enough (but constant) $\varepsilon > 0$, we cannot have a poly-time $\left(O\left(2^{c\sqrt{\log N}}\right), 1-\varepsilon\right)$-bicriteria approximation algorithm for the* `MaxStaff` *problem, unless $NP \subseteq DTIME\left(n^{O(\log n)}\right)$. Here N denotes the size of the input instance, and c is a constant (depending on ε).*

Our Techniques One can write a "natural" LP relaxation for the `MaxStaff` problem. However, this turns out to have unbounded gap. We first show that if we are willing to lose a constant factor (depending on ε) in the profit, then there is a significant amount of structure in any solution. We first consider the special case when all the requirements are same : this happens to capture many of the difficulties. First, we divide the timeline into *blocks* : each block is a sequence of L continuous time slots. Then we show that, with a factor 2 loss in approximation, we can assume that each selected interval contributes to satisfying the requirements of only one block. Thus, we can decompose a solution into several independent solutions, one for each block (of course, we must satisfy the global requirement that we pick only α_i intervals from $\mathcal{E}_i$). We then focus on a particular block, and think of the solution restricted to just this block. We further simplify the structure of a solution : we show that we can find a continuous set of timeslots in a block, which we call a sub-block, and select a subset of intervals in the solution such that the following conditions hold : (i) each interval in the latter solution contains the entire sub-block and satisfy the requirements of

the timeslots in this sub-block (upto an ε factor loss in the requirement), and (ii) the total profit accrued from this sub-block is at least a constant times the total profit obtained by the solution from the entire block. This simple structure allows us to write a configuration LP – we show that we can solve this LP and a simple randomized rounding based algorithm gives a good approximation algorithm. The extension to the case of arbitrary requirements is based on standard geometric grouping ideas.

For the hardness result, the reduction from the `Maximum Independent Set` problem follows in a straight-forward manner. Using this one can also show that if we want a bicriteria approximation for a parameter ε, then there is a constant (depending on ε) hardness. For the case when the lengths of the intervals in the input can vary (significantly), we show that this construction can be recursively composed to amplify the hardness of the problem to that mentioned in Theorem 3.

Organization of the Paper In Section 2, we define the problem formally and give some related definitions. In Section 3, we prove both the hardness results. In Section 4, we describe the bi-criteria approximation algorithm. In Section 4.1, we first consider the case when all the requirements r_t are same. This special case captures many of the ideas, and simplifies our presentation. In Section 4.2, we outline the ideas needed to extend this to the case of arbitrary requirements. The details of the algorithm for the latter case are deferred to the full version.

2 Problem Definition and Notation

We define the `MaxStaff` problem. An input instance $\mathcal{I}$ is specified by a tuple $(T, R, \mathcal{E}, r, \alpha, p, L)$. The elements of T correspond to consecutive time slots, and they are numbered in this order from 1 to $|T|$. The set R is the set of available resources, $\mathcal{E}$ is a mapping from R to the power set of all intervals of length L in T. We denote by $\mathcal{E}_i$ the set of intervals (of length L) associated with $i \in R$. The quantities r, p, α are mappings from the sets T, T, R to the set of positive integers respectively. The value r_t denotes the requirement of timeslot t, p_t denotes its profit, and α_i denotes the "availability" of resource i, i.e., the maximum number of intervals we may pick from the set $\mathcal{E}_i$.

We say that two intervals I, I' in the input are *mutually distinct* if they belong to different sets $\mathcal{E}_i$. A solution $\mathcal{S}$ selects for each $i \in R$, a subset $\mathcal{E}_i(\mathcal{S})$ of at most α_i intervals from $\mathcal{E}_i$. Let $\mathcal{E}(\mathcal{S})$ denote $\cup_i \mathcal{E}_i(\mathcal{S})$. We say that $t \in T$ gets *satisfied* by $\mathcal{S}$ if there are at least r_t mutually distinct intervals containing t in the set $\mathcal{E}(S)$. Note that the intervals in $\mathcal{E}_i(\mathcal{S})$ can overlap, but they contribute at most once towards the number of mutually distinct intervals containing any timeslot. The goal is to maximize the total profit of time slots which get satisfied.

For the hardness result, the length L may vary with the resource i – for each $i \in R$, the input instance specifies a length L_i. So all intervals in $\mathcal{E}_i$ are of length L_i.

3 Hardness Results

3.1 Proof of Theorem 1

We reduce the `Maximum Independent Set` problem to the `MaxStaff` problem. Consider an instance $\mathcal{I}' = (G = (V, E))$ for the `Maximum Independent Set` problem. We construct an instance $\mathcal{I} = (T, R, \mathcal{E}, r, \alpha, p, L)$ of the `MaxStaff` problem as follows. Let $V = \{v_1, \ldots, v_n\}$. The set T will be equal to $\{1, \ldots, |V|\}$ and a timeslot $t \in T$ will correspond to the vertex v_t. For each edge $e \in E$, we have an element $i_e \in R$. We now describe the set $\mathcal{E}$ in $\mathcal{I}$. For an edge $e = (v_a, v_b) \in E$, the set $\mathcal{E}_{i_e}$ contains two intervals – $\{a\}, \{b\}$. Note that the parameter L is equal to 1. For every $t \in T$, the requirement r_t is equal to the degree of the vertex v_t in G. Further, we have $\alpha_i = 1$ for all $i \in R$, and $p_t = 1$ for all $t \in T$ (see Figure 1 for an example).

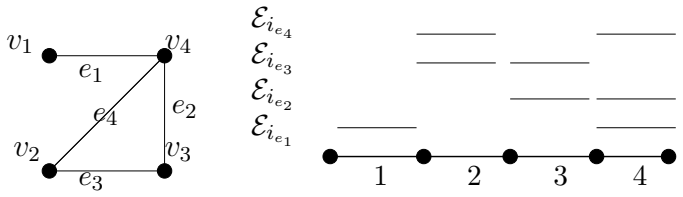

Fig. 1. Example showing the reduction from `Maximum Independent Set` to the `MaxStaff` problem.

Claim. There is a solution to the `MaxStaff` problem instance $\mathcal{I}$ of profit at least k iff there is an independent set in G' of size k.

Proof. Suppose there is an independent set S of size k in G. Let $T' = \{t : v_t \in S\}$ be the corresponding timeslots in T (in the instance $\mathcal{I}$). Consider the following solution : for any edge $e = (v_a, v_b) \in E$, at most one of the vertices in v_a and v_b belongs to S – assume wlog that $v_a \in S$. Then we pick the interval $\{a\} \in \mathcal{E}_{i_e}$ (in case none of these vertices is in S, we pick any one of the two intervals in $\mathcal{E}_{i_e}$). It is easy to check that this solution satisfies the requirements of all the timeslots in T'.

Conversely, suppose that there is a solution for $\mathcal{I}$ which satisfies k timeslots in T – call this set T'. Again, it is easy to check that the set $S = \{v_t : t \in T'\}$ is an independent set in G. □

This concludes the proof of Theorem 1.

3.2 Proof of Theorem 3

Fix a small enough constant $\varepsilon > 0$. Assume wlog that $\frac{1}{\varepsilon}$ is an integer and let Δ denote this quantity. We use the following result of Samorodnitsky and Trevisan [ST00] which shows hardness of the `Maximum Independent Set` problem on bounded degree graphs.

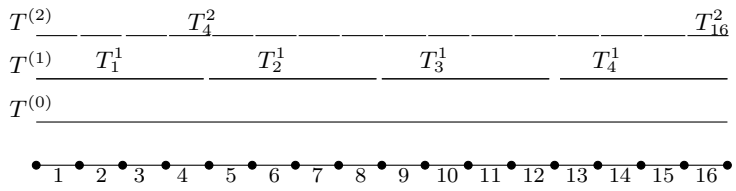

Fig. 2. The sets of intervals in $T^{(0)}$ and $T^{(1)}$ for the graph in Figure 1.

Theorem 4. *[ST00] Unless P=NP, there is no poly-time* $\frac{\Delta}{2^{O(\sqrt{\log \Delta})}}$*-approximation algorithm for the* `Maximum Independent Set` *problem on graphs of bounded degree* Δ*.*

We start with an instance $\mathcal{I}' = (G = (V, E))$ of the `Maximum Independent Set`-problem, where the maximum degree of a vertex in G is Δ. For ease of notation, we can in fact assume by adding self-loops that the degree of each vertex is exactly Δ – we modify the definition of an independent set as a set of vertices such that for any edge, at most *one* of its end-points is in this set.

We now construct an instance $\mathcal{I} = (T, R, \mathcal{E}, r, \alpha, p, L)$ of the `MaxStaff` problem. Recall that L_i denotes the length of the intervals in $\mathcal{E}_i$. The idea behind the proof is the following : we start with the reduction from `Maximum Independent Set` to `MaxStaff` as described in the previous section. Now we stretch each timeslot n times, i.e., we replace it by n consecutive timeslots (so the intervals which were earlier of length 1 now have length n). For each of these newly created n timeslots (corresponding to one timeslot in the original instance), we again embed the `MaxStaff` instance used in the reduction. This process is continued recursively for K stages. We now give details of our construction.

The set T will have size n^K, where $n = |V|$ in the instance $\mathcal{I}'$ and K is a parameter to be chosen later. We shall construct the set of elements in R and the set $\mathcal{E}$ in several stages. We first divide the timeline T into a laminar family of sets. For each value of k between 0 and K, we do the following : divide the set T (starting from the first timeslot) into disjoint intervals of n^{K-k} consecutive points each – call these intervals $T_1^k, \ldots, T_{r_k}^k$, where $r_k = n^k$. Let $T^{(k)}$ denote the set of these intervals. It is easy to check that T_r^k contains n intervals from the set $T^{(k+1)}$ and is disjoint from other intervals in $T^{(k+1)}$ (see Figure 2 for an example with $K = 2$).

We now construct the sets R and $\mathcal{E}$. For each value of k, $0 \leq k \leq K-1$, and $1 \leq r \leq n^k$, we add a set of elements $R(k, r)$ to the set R. Recall that T_r^k contains n intervals from the set $T^{(k+1)}$ – call these intervals $T_{s_1}^{k+1}, \ldots, T_{s_n}^{k+1}$. Let the vertices in V be $v_1, \ldots, v_n$ (recall that V is the set of vertices in the `Maximum Independent Set` instance $\mathcal{I}'$). For each edge $e = (v_a, v_b) \in E$, we add an element $e(k, r)$ to $R(k, r)$. Further, we have two intervals in $\mathcal{E}_{e(k,r)}$: $T_{s_a}^{k+1}$ and $T_{s_b}^{k+1}$ – we denote these intervals by $I^1(e(k, r))$ and $I^2(e(k, r))$. Notice that there are exactly Δ intervals of the form $I^l(e(k, r))$, $l = 1, 2$, $e \in E$, which are equal

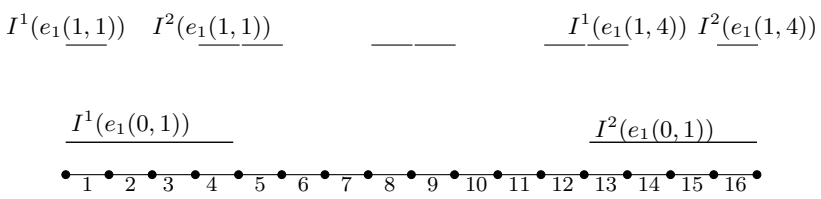

Fig. 3. The intervals corresponding to $I^l(e_1(k,r))$, where $l = 1, 2$, $k = 0, 1$ and $1 \leq r \leq n^k$, and the input graph corresponds to the instance in Figure 1.

to $T^{(k+1)}_{s_a}$, $1 \leq a \leq n$. Note that $L_i = n^{K-k}$ for all $i \in R(k,r)$. This completes the description of how $R(k,r)$ and the corresponding sets $\mathcal{E}_i, i \in R(k,r)$ are constructed (see Figure 3 for an example). We set the values α_i equal to 1 for all $i \in R$, $p_t = 1$ for all $t \in T$ and $r_t = K \cdot \Delta$ for all $t \in T$.

In the following analysis, it will be convenient to associate a K-tuple B^t with any timeslot $t \in T$ as follows : for a fixed k, $0 \leq k \leq K-1$, let r be such that $t \in T^k_r$. As before, there are n intervals from $T^{(k+1)}$ which are contained in T^k_r – call these intervals $T^{k+1}_{s_1}, \ldots, T^{k+1}_{s_n}$ from left to right. Then the coordinate for $(k+1)$ in the vector B^t, i.e., $B^t(k+1) = l$ if $t \in T^{k+1}_{s_l}$. Observe that this vector gives the sequence of nested intervals which contain this timeslot.

Lemma 1. *If $\mathcal{I}'$ has an independent set of size A, then there is a solution to $\mathcal{I}$ of profit A^K.*

Proof. Let S be such an independent set of size A. We construct a solution for $\mathcal{I}$. For each element $e(k,r) \in R$, where $e = (v_a, v_b) \in E, 0 \leq k \leq K-1, 1 \leq r \leq r_k$, we need to pick one of the two intervals – $T^{k+1}_{s_a}$ and $T^{k+1}_{s_b}$, where s_a, s_b are the a^{th} and the b^{th} intervals of $T^{(k+1)}$ contained inside T^k_r respectively. If $v_b \notin S$, then we pick $T^{k+1}_{s_a}$, else we pick $T^{k+1}_{s_b}$. It is easy to check that in such a solution, all elements t for which $B^t = (a_1, \ldots, a_K), v_{a_i} \in S$ have $\Delta \cdot K$ distinct intervals containing it. This proves the lemma. □

Let γ denote the parameter $\frac{2^{O(\sqrt{\log \Delta})}}{\Delta}$ which appears in Theorem 4. Suppose there is no independent set of size $\gamma \cdot A$ in $\mathcal{I}'$. We first prove a technical lemma below.

Lemma 2. *Suppose we partition the edge set E into n sets — E_v for every $v \in V$, where E_v is a subset of edges incident with v. For a constant c, $0 \leq c \leq \Delta$, let $W(c) \subseteq V$ be those vertices v for which $|E_v| \geq \Delta - c$. Then, $|W(c)| \leq \gamma \cdot A \cdot (2c+1)$.*

Proof. We prove that G has an independent set of size at least $\frac{|W(c)|}{2c+1}$. In fact, this independent set will be a subset of $W(c)$. We prove this fact as follows : consider any subset $S \subseteq W(c)$, and let $G[S]$ be the subgraph of G induced by S. Then there must be a vertex in $G[S]$ of degree at most $2c$. Once we prove this fact, it is easy to check that a simple greedy algorithm will yield the desired independent set.

So fix a set $S \subseteq W(c)$. We direct each edge in $G[S]$ as follows. Suppose $e = (v, w) \in G[S]$. If $e \in E_v$, then we direct e from v to w, otherwise ($e \in E_w$) we direct it from w to v. Since every $v \in S$ is also in $W(c)$, the indegree of any vertex is at most c. So there are at most $|S|c$ edges in $G[S]$, which implies that there must be a vertex of (undirected) degree at most $2c$ in $G[S]$. □

We are now ready to prove that $\mathcal{I}$ cannot have a solution of high profit.

Theorem 5. *Let $\mathcal{S}$ be a solution to $\mathcal{I}$ and $T^{\mathcal{S},\varepsilon}$ be the timeslots in T which are at least $(1-\varepsilon)$-satisfied. Then $|T^{\mathcal{S},\varepsilon}| \leq (10\gamma \log \Delta)^K A^K$.*

Proof. Recall that $\mathcal{E}(\mathcal{S})$ denotes the set of intervals selected by $\mathcal{S}$ – this set contains exactly one element from $\mathcal{E}_i$ for any $i \in R$. If $t \in T^{\mathcal{S},\varepsilon}$, then there are at least $\Delta \cdot K - K$ intervals in $\mathcal{E}(\mathcal{S})$ containing t. Recall that the instance $\mathcal{I}$ has exactly $\Delta \cdot K$ intervals containing t. We can associate a tuple of size K with t, which we call $Z^{\mathcal{S},t}$ as follows : fix a value k between 1 and K. Let r be such that $t \in T_r^{k-1}$. Then there are exactly Δ intervals from the sets $\mathcal{E}_{e(k,r)}, e \in E$, which contain t (and are therefore, contained in T_r^{k-1}). Among these Δ intervals, suppose $\mathcal{S}$ chooses $\Delta - c_k$ intervals. Then define $Z^{\mathcal{S},t}(k) = c_k$. Observe that $\sum_k Z^{\mathcal{S},t} \leq K$ for each t. We now define a *truncated* version $Y^{\mathcal{S},t}$ of the vector $Z^{\mathcal{S},t}$ as follows. $Y^{\mathcal{S},t}(k)$ is the nearest power of 2 greater than or equal to $Z^{\mathcal{S},t}(k)$ (unless $Z^{\mathcal{S},t}(k)$ is 0, in which case, $Y^{\mathcal{S},t}(k)$ stays as 0). We first count how many different values the vector $Y^{\mathcal{S},t}$ can take.

Claim. The number of distinct values (as a vector) that $Y^{\mathcal{S},t}$ can take is at most $(\log \Delta + 2)^K$.

Proof. Each coordinate of $Y^{\mathcal{S},t}$ is a power of 2 (or 0), and is at most 2Δ. So the number of distinct values it can take is at most $\log(2\Delta) + 1$. □

Now, we bound the number of different $t \in T^{\mathcal{S},\varepsilon}$ for which $Y^{\mathcal{S},t}$ takes a particular (vector) value.

Claim. Let $c_1, \ldots c_K$ be such that each c_i is either 0 or a power of 2, and $\sum_{k=1}^{K} c_k \leq 2K$. Then, the number of different $t \in T^{\mathcal{S},\varepsilon}$ for which $Y^{\mathcal{S},t} = (c_1, \ldots, c_K)$ is at most $5^K \gamma^K A^K$.

Proof. For each k, $1 \leq k \leq K$, let $r(k,t)$ be such that $t \in T^k_{r(k,t)}$. Since, $Z^{\mathcal{S},t}(k) \leq c_k$, Lemma 2 implies that for any fixed values of $T^{k'}_{r(k',t)}$ for all $k' < k$, there are at most $\gamma A \cdot (2c_k + 1)$ possibilities for $T^k_{r(k,t)}$. So the number of possible t satisfying the condition of the claim is at most $(\gamma A)^K \cdot \Pi_{k=1}^K (2c_k + 1)$. The quantity in the product (subject to $\sum_k c_k \leq 2K$) is at most 5^K. □

The above two claims now imply the theorem. □

Lemma 1 and Theorem 5, along with Theorem 4 show that it is NP-hard to get a $\left(\frac{\Delta}{2^{O(\sqrt{\Delta})}}\right)^K$-approximation for the `MaxStaff` problem even we relax the notion of satisfiability to $(1-\varepsilon)$-satisfiability for our algorithm. We now set $K = \log n$. So, the size of $\mathcal{I}$ is roughly $N = n^{\log n}$ and hence, the hardness becomes $\Omega(2^{c\sqrt{\log N}})$.

4 The bicriteria approximation algorithm

In this section, we prove Theorem 2. It is not difficult to show the natural LP relaxation for this problem has a large integrality gap. We, therefore, write a stronger *configuration* LP for this problem. A configuration LP, by definition, has (perhaps exponential number of) variables which encode all possible *configurations* of a solution : these configurations should be rich enough to capture essential properties required for rounding, and yet one should be able to solve the LP in polynomial time. Getting the right balance between these two aspects turns out to be non-trivial in our setting. We motivate the ideas behind the configuration LP by looking at the following special case : requirements of all timeslots are same, i.e., $r_t = r$ for all $t \in T$. Although it requires some technical work to extend this to the general case, this special case illustrates many of the difficulties involved in coming up with a configuration LP.

4.1 The Special Case of Uniform Requirements

We consider the special case when $r_t = r$ for all timeslots. Let $\mathcal{S}$ be an optimal solution. Recall that $\mathcal{E}(S)$ denotes the set of intervals selected by $\mathcal{S}$.

Definition 1. *A* block B *is a set of* L *consecutive timeslots in* T*. We divide the timeline into* $N = \lceil \frac{T}{L} \rceil$ *blocks, namely,* $B_1, \ldots, B_N$*, in a natural manner – block* B_k *consists of timeslots* $\{(k-1)L+1, (k-1)L+2, \ldots, (k-1)L+L\}$ *(except perhaps the last block which may end early).*

To begin with, we would like to consider only a subset of blocks.

Definition 2. *We say that a set of blocks* $\mathcal{B}$ *is* independent *if no interval in* $\mathcal{E}(S)$ *intersects more than one block in* $\mathcal{B}$*. Let* $\mathcal{B}_o$ *be the set of odd numbered blocks :* $B_1, \mathcal{B}_3, \ldots$ *and* $\mathcal{B}_e$ *be the even numbered blocks : it is easy to check that* $\mathcal{B}_o$ *(or* $\mathcal{B}_e$*) is an independent set of blocks (see Figure 4).*

Since the total profit of timeslots satisfied by $\mathcal{S}$ in $\mathcal{B}_o$ or $\mathcal{B}_e$ is at least half the total profit of satisfied timeslots, we can focus (with a factor-2 loss in approximation) on an independent set of blocks – call this $\mathcal{B}$ (so, $\mathcal{B}$ is either $\mathcal{B}_e$ or $\mathcal{B}_o$). Fix a block $B \in \mathcal{B}$.

Definition 3. *A sub-block,* sB*, of* B *is a set of continuous time-slots in* B*.*

Definition 4. *Let* $\texttt{profit}(\mathcal{S}, B)$ *be the total profit of timeslots in* B *which are satisfied by* $\mathcal{S}$*. Given a sub-block* sB *of* B*, and a solution* $\mathcal{S}'$*, let* $\texttt{profit}^{\varepsilon}(\mathcal{S}', sB)$ *denote the total profit of timeslots in* sB *which are* $(1-\varepsilon)$*-satisfied by* $\mathcal{S}'$*.*

Lemma 3. *Consider a block* B *and a solution* $\mathcal{S}$*. There exists a sub-block* sB *of* B *and a solution* $\mathcal{S}'$ *consisting of a subset of intervals selected by* $\mathcal{S}$ *such that the following conditions are satisfied :*

- *The intervals in* $\mathcal{S}'$ *are mutually distinct.*

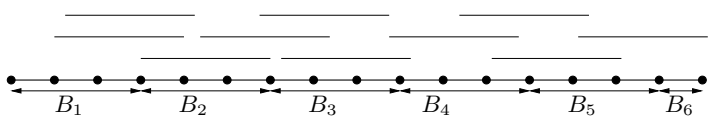

Fig. 4. The set of blocks when $L = 3$. Here, $\mathcal{B}_o = \{B_1, B_3, B_5\}, B_e = \{B_2, B_4, B_6\}$. Note that any interval picked by a solution intersects at most one block in $\mathcal{B}_o$ (or $\mathcal{B}_e$).

- *Every interval in $\mathcal{S}'$ contains the sub-block sB.*
- $\texttt{profit}^{\varepsilon}(\mathcal{S}', sB) \geq \varepsilon \cdot \texttt{profit}(\mathcal{S}, B)$.

Proof. Let $\mathcal{S}^{\texttt{left}}$ (or $\mathcal{S}^{\texttt{right}}$) be the intervals in $\mathcal{S}$ which contain the left (or right) end-point of B – we can assume wlog that the sets $\mathcal{S}^{\texttt{left}}$ and $\mathcal{S}^{\texttt{right}}$ are disjoint (any common interval is assigned to just one of the two sets arbitrarily). Also, we can assume wlog that the intervals in $S^{\texttt{left}}$ (or $S^{\texttt{right}}$) are mutually distinct, i.e., they contain at most one interval from $\mathcal{E}_i$ for any resource $i \in R$. Indeed, we can keep the interval in $S^{\texttt{left}} \cap \mathcal{E}_i$ which goes farthest to the right and remove the rest from this set.

Let B' be the timeslots in B which are $(1-\varepsilon)$-satisfied by $\mathcal{S}$ – we denote these from left to right as as $t_1, \ldots, t_k$. We now define $\tau = \frac{1}{\varepsilon}$ disjoint sub-blocks $sB_1, sB_2, \cdots, sB_\tau$ of B. Let n_i, $1 \leq i \leq \tau$ be the smallest integer such that the number of intervals from $\mathcal{S}^{\texttt{left}}$ containing t_{n_i} is at most $(1 - i\varepsilon)r$. Define n_0 as 1. Let sB_i denote the interval $[t_{n_{i-1}}, t_{n_i})$. Consider any sub-block sB_i. Let $\mathcal{S}_1 \subseteq \mathcal{S}^{\texttt{right}}$ be the intervals which contain $t_{n_{i-1}}$ and $\mathcal{S}_2 \subseteq \mathcal{S}^{\texttt{left}}$ be those intervals which contain t_{n_i}. Let $\mathcal{K}$ denote $\mathcal{S}_1 \cup \mathcal{S}_2$. Clearly, the intervals in $\mathcal{K}$ contain sB_i, and must have at least $r(1-\varepsilon)$ distinct intervals. We are now done because we can choose the sub-block sB_i for which $\texttt{profit}^{\varepsilon}(\mathcal{S}, sB_i)$ is maximized. □

The configuration LP We are now ready to write the configuration LP. The configurations will be defined by pairs $(B, \mathcal{A})$, where B is a block and $\mathcal{A}$ is a set of mutually distinct intervals each of which has a non-empty intersection with B. Motivated by Lemma 3, we have the following definition.

Definition 5. *For a block B, and a set of intervals $\mathcal{A}$, let $p(\mathcal{A}, B)$ denote the maximum over all sub-blocks sB of B of the quantity $\texttt{profit}^{\varepsilon}(\mathcal{A}', sB)$, where $\mathcal{A}'$ contains those intervals in $\mathcal{A}$ which contain the entire sub-block sB.*

The configuration LP has variables $y(\mathcal{A}, B)$ which is 1 when we pick all the intervals in $\mathcal{A}$, 0 otherwise.

$$\begin{aligned} \max \quad & \sum_{B \in \mathcal{B}} \sum_{\mathcal{A}} y(\mathcal{A}, B) \cdot p(\mathcal{A}, B) \\ \sum_{\mathcal{A}: \mathcal{A} \cap \mathcal{E}_i \neq \emptyset} & y(\mathcal{A}, B) \leq \alpha_i \qquad \text{for every } i \in R \\ \sum_{\mathcal{A}} & y(\mathcal{A}, B) \leq 1 \qquad \text{for every } B \\ & y(\mathcal{A}, B) \geq 0 \qquad \text{for all } \mathcal{A}, B \end{aligned} \tag{1}$$

It is not difficult to show that there is a polynomial time separation oracle for the dual of this LP, and so, it can be solved in polynomial time. The details are deferred to the full version.

Rounding a fractional solution Let $y(\mathcal{A}, B)$ be an optimal solution to the configuration LP. We now show that it can be rounded to an integral solution without much loss in profit. Assume that the LP is written for $\mathcal{B} = \mathcal{B}_o$. Consider the algorithm in Figure 5. Observe that we can perform the sampling required in Step 1 because of constraint (1).

For each block $B \in \mathcal{B}_o$ do

1. Use dependent rounding to sample at most *one* of the set of intervals $\mathcal{A}$, where the probability of picking $\mathcal{A}$ is exactly $\varepsilon \cdot y(\mathcal{A}, B)$.
2. Suppose we select a set of intervals $\mathcal{A}$ in the experiment above (otherwise skip to the next step).
3. For every interval $I \in \mathcal{A}$ do
   Remove I from the set of selected intervals if we have already selected α_i intervals from the set $\mathcal{E}_i$ containing I.

Output the set of selected intervals.

Fig. 5. Algorithm `Round`

Theorem 6. *The total profit of timeslots which are* $(1 - 3\varepsilon)$*-satisfied by the solution output by the algorithm* `Round` *is at least* $\Omega(\varepsilon)$ *times the objective value of the fractional solution* $y(\mathcal{A}, B)$*.*

Proof. Note that an interval is selected only if the following two conditions hold: (i) it is picked as a part of some $\mathcal{A}$ in Step 1 of the above algorithm, and (ii) it is not dropped in Step 3 of the above algorithm. Given a set of intervals $\mathcal{A}$ picked for a specific B, the profit $p(\mathcal{A}, B)$ is accrued from some sub-block sB of B where the timeslots in sB are $(1-\varepsilon)$-satisfied by the intervals in $\mathcal{A}$. However, because of Step 3 of the algorithm, some of the intervals in $\mathcal{A}$ may get *dropped*. Each interval in $\mathcal{A}$ gets dropped (by Step 1) with a probability of at most ε. This holds even when α_i intervals are to be selected for resource i (the probability may be lower since choices of $\mathcal{A}$ according to $y(\mathcal{A}, B)$ are independent across the various blocks). Thus the expected fraction of intervals dropped is at most ε. By Markov's inequality, the probability that the number of intervals dropped is $\geq 2\varepsilon$ is at most $1/2$. Thus with probability at least $1/2$, a timeslot in sB is $(1 - \varepsilon - 2\varepsilon) = (1 - 3\varepsilon)$-satisfied by the modified set $\mathcal{A}$. Given the choice of the sets of intervals $\mathcal{A}$, it is clear from the above that (in expectation) the profit is at least $\varepsilon/2$ times the objective value of the LP. □

4.2 Extending to General Requirements

We briefly describe how we generalize the above algorithm to the case when the requirements r_t are arbitrary. As before, we can focus on a set of independent blocks $\mathcal{B}$. We first assume that the requirement of any timeslot, r_t, is a power

of $(1+\varepsilon)$. We achieve this by rounding r_t values down to the nearest power of $(1+\varepsilon)$. Since this only affects the r_t value by at most εr_t, and we are only interested in a bicriteria approximation, this assumption is valid.

Definition 6. *We say that a timeslot t is of* class c *if $r_t = (1+\varepsilon)^c$. Let $T(c)$ denote the timeslots of class c. A set T' of timeslots is said to be* well-separated *if for any two $t_1, t_2 \in T'$, where t_1 is of class c_1 and t_2 is of class c_2, either $c_1 = c_2$ or $c_1 - c_2 \geq H$. Here H is a parameter equal to $\frac{2}{\varepsilon} \cdot \ln \frac{1}{\varepsilon}$.*

We can assume that the set of timeslots are well-separated (upto a constant loss in approximation) – indeed, for a value h between 0 and $H-1$, consider only those timeslots whose class is equal to h modulo H. For any fixed value of h, these timeslots are well-separated. Since h takes only H distinct values, at least one of these values will give us at least $1/H$ fraction of the profit. So we assume that the timeslots are well-separated and let $\mathcal{C}$ denote the set of possible classes of these timeslots.

Now, given any solution $\mathcal{S}$, we would like to divide the intervals in $\mathcal{S}$ into $|\mathcal{C}|$ disjoint sets — $\mathcal{S}^c, c \in \mathcal{C}$, such that if a timeslot of class c gets satisfied by $\mathcal{S}$, then it will be at least $(1-\varepsilon)$-satisfied by $\mathcal{S}^c$. This will allow us to think of any solution as $|\mathcal{C}|$ independent solutions, one for each class $c \in \mathcal{C}$. Once we have this, then we can use the machinery developed (for uniform requirements) in the previous section for each class separately and combine the solutions. We now state this key theorem more formally. Fix a block B, and let B^c be the timeslots in B of class c. For a solution $\mathcal{S}$, define $\texttt{profit}(\mathcal{S}, \mathcal{B}^c)$ as the total profit of timeslots in $\mathcal{B}^c$ satisfied by $\mathcal{S}$. Define $\texttt{profit}^\varepsilon(\mathcal{S}, \mathcal{B}^c)$ similarly.

Theorem 7. *Fix a block B. Given a solution $\mathcal{S}$, we can partition the intervals in this solution to get solutions $\mathcal{S}^c, c \in \mathcal{C}$, such that*

$$\sum_{c \in \mathcal{C}} \texttt{profit}^\varepsilon(\mathcal{S}^c, B^c) \geq \texttt{profit}(\mathcal{S}, \cup_{c \in \mathcal{C}} B^c).$$

The proof of the theorem essentially uses the fact that there is a large gap between the requirements of timeslots belonging to two different classes. We defer the proof to the full version. Once we have this, we use the structural property in Lemma 3 for each class $c \in \mathcal{C}$ separately. This allows us to write a configuration LP. We round this LP using ideas similar to the uniform requirement case.

5 Discussion

Our algorithm can be easily extended to take care of following cases :

- Different resources may have different *proficiencies* – we say that a timeslot t is satisfied if there are a set of mutually distinct selected intervals whose total proficiency exceeds r_t. Again, we can get the same bicriteria approximation algorithm for this problem.
- Let β denote the ratio of the maximum length of an interval in $\cup_{i \in R} \mathcal{E}_i$ to the smallest length. Then we can get an $\left(\frac{\beta}{\varepsilon^3} \cdot \log \frac{1}{\varepsilon}, 1-\varepsilon\right)$-bicriteria approximation algorithm for this problem. This allows a resource to have different shift

lengths depending on when it is scheduled. The only thing that changes in the algorithm is that initially we divide the timeslots into blocks of length $L_{\min}$, where $L_{\min}$ is the smallest length of an interval. Now, instead of just looking into sets $\mathcal{B}_o$ and $\mathcal{B}_e$, we divide these blocks into β classes.
- We may stipulate that any two picked intervals from the same set $\mathcal{E}_i$ must have enough separation Δ between them; this makes sense because two different shifts of an employee should not be too close to each other. Our algorithm remains the same, except that this constraint is added in the configuration LP (and the randomized rounding procedure changes accordingly).

There are many interesting directions in which this work can be extended. In the context of contact centers, people with various *skills* may be required (say, knowledge of various languages). For a small set K of skills, an employee may have a subset of these skills. Now the requirement at time t may specify the minimum number of employees of each skill required. Modeling such problems and coming up with good approximation algorithms for them is a challenging problem. Further, instead of considering the problem of maximizing the number of timeslots satisfied, we can also consider the problem of minimizing the number of employees such that all timeslots are satisfied. However, checking feasibility of such an instance (whether all employees can be scheduled to satisfy all the timeslots) itself is NP-complete. Hence, it will be interesting to find tractable special cases of this problem. Finally, the requirements themselves are estimates. Posing these problems in a stochastic setting is also a very challenging problem.

References

[AAM07] Zeynep Aksin, Mor Armony, and Vijay Mehrotra. The modern call center: A multi-disciplinary perspective on operations management research, 2007.

[CCPV07] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint (extended abstract). In *IPCO*, pages 182–196, 2007.

[FHF+02] Alex Fukunaga, Ed Hamilton, Jason Fama, David Andre, Ofer Matan, and Illah Nourbakhsh. Staff scheduling for inbound call centers and customer contact centers. In *AAAI*, pages 822–829, 2002.

[GKM03] Noah Gans, Ger Koole, and Avishai Mandelbaum. Telephone call centers: Tutorial, review, and research prospects. *MSOM*, 5(2):79–141, 2003.

[GRST10] Anupam Gupta, Aaron Roth, Grant Schoenebeck, and Kunal Talwar. Constrained non-monotone submodular maximization: Offline and secretary algorithms. In *WINE*, pages 246–257, 2010.

[ICWC10] Armann Ingolfssona, Fernanda Campelloa, Xudong Wub, and Edgar Cabralc. Combining integer programming and the randomization method to schedule employees. *European Journal of Operations Research*, 2010.

[LMNS09] Jon Lee, Vahab S. Mirrokni, Viswanath Nagarajan, and Maxim Sviridenko. Non-monotone submodular maximization under matroid and knapsack constraints. In *STOC*, pages 323–332, 2009.

[Rey03] Penny Reynolds. *Call Center Staffing: The Complete, Practical Guide to Workforce Management*. The Call Center School Press, 2003.

[ST00] Alex Samorodnitsky and Luca Trevisan. A PCP characterization of NP with optimal amortized query complexity. In *STOC*, pages 191–199, 2000.