

# A Constant Factor Approximation Algorithm for Generalized Min-Sum Set Cover

Nikhil Bansal\*

Anupam Gupta<sup>†</sup>

Ravishankar Krishnaswamy<sup>†‡</sup>

## Abstract

Consider the following *generalized min-sum set cover* or *multiple intents re-ranking* problem proposed by Azar et al. (STOC 2009). We are given a universe of elements and a collection of subsets, with each set  $S$  having a covering requirement of  $K(S)$ . The objective is to pick one element at a time such that the average *covering time* of the sets is minimized, where the covering time of a set  $S$  is the first time at which  $K(S)$  elements from it have been selected.

There are two well-studied extreme cases of this problem: (i) when  $K(S) = 1$  for all sets, we get the min-sum set cover problem, and (ii) when  $K(S) = |S|$  for all sets, we get the minimum-latency set cover problem. Constant factor approximations are known for both these problems. In their paper, Azar et al. considered the general problem and gave a logarithmic approximation algorithm for it. In this paper, we improve their result and give a simple randomized constant factor approximation algorithm for the generalized min-sum set cover problem.

## 1 Introduction

The *min-sum set cover* problem is a min-latency version of the well-known set cover problem: for ease of exposition we will consider the equivalent hitting set formulation of the set cover problem. Here, we are given a universe  $U$  of  $n$  elements, and a collection  $\mathcal{S} = \{S_1, S_2, \dots, S_m\}$  of subsets with  $S_i \subseteq U$ , and the objective is to select one element at a time (i.e., find a linear ordering of the elements) such that the average hitting (or “cover”) time of the sets is minimized. Formally, we pick one element at every time instant: if an element  $e$  is picked at time  $t$  its cover time is  $\text{Cov}(e) = t$ . The hitting/cover time

of a set  $S$  is  $\text{Cov}(S) = \min_{e \in S} \text{Cov}(e)$ , and the goal is to minimize  $\sum_{S \in \mathcal{S}} \text{Cov}(S)$ . For this problem the greedy algorithm of picking the element which covers the most number of uncovered sets is known to be a 4-approximation for this problem [BNBH<sup>+</sup>98, FLT04], and this is the best possible unless  $P=NP$  [FLT04]. A problem that is similar in spirit is the *min-latency set cover* problem, where the cover time of a set  $S$  is  $\text{Cov}(S) = \max_{e \in S} \text{Cov}(e)$ , the time at which *all* the elements of the set have been selected. This problem also admits a constant factor approximation algorithm [HL05]. In fact, this problem easily reduces to that of precedence-constrained scheduling on a single machine, for which a 2-approximation is known using various techniques [HSSW97, MQW03, CM99].

A substantial generalization of these two problems was offered recently by Azar, Gamzu and Yin [AGY09]: the *multiple intents re-ranking problem* or the *generalized min-sum set cover problem* (GenMSSC). Here each set  $S \in \mathcal{S}$  also comes with a covering requirement  $K(S) \in \{1, 2, \dots, |S|\}$ , and its cover time is defined to be the time at which  $K(S)$  elements from  $S$  are selected:

$$\text{Cov}(S) = \min\{t \mid \#(e \in S \text{ s.t. } \text{Cov}(e) \leq t) = K(S)\}.$$

The goal is to minimize  $\sum_S \text{Cov}(S)$ . Note that we get the min-sum set cover problem if we set  $K(S) = 1$  for all sets  $S \in \mathcal{S}$ , and the min-latency set cover problem if we set  $K(S) = |S|$  for all  $S \in \mathcal{S}$ . Azar et al. [AGY09] gave an  $O(\ln r)$ -approximation algorithm for this problem, where  $r$  is the largest size of any set in  $\mathcal{S}$  via a modified greedy algorithm, and left open the question of obtaining a constant factor approximation for the problem. We resolve that question in this paper.

**THEOREM 1.1.** *The generalized min-sum set cover problem (a.k.a the multiple intents re-ranking problem) admits a randomized 485-approximation algorithm.*

Our approach is based on formulating a strengthened LP relaxation for the problem, obtained by adding the so-called “knapsack-cover inequalities” [CFLP00] to the natural LP relaxation. This is necessary as one can construct examples (see Section 6.2) where the natural

\*IBM T. J. Watson Research Center, Yorktown Heights, NY 10598.

<sup>†</sup>Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, USA. Supported in part by NSF awards CCF-0448095 and CCF-0729022, and an Alfred P. Sloan Fellowship.

<sup>‡</sup>Research partly done while visiting IBM T. J. Watson Research Center.

LP has an unbounded integrality gap. We then use a simple stage-based randomized rounding scheme which works as follows. We consider exponentially increasing prefixes of time, and round the (fractional) assignments in these prefixes to obtain partial orderings. Then, we combine these partial orderings into a single ordering. For any set  $S$ , our rounding guarantees an expected cover time of  $O(t_S)$ , where  $t_S$  is its cover time in the LP relaxation.

**1.1 Related Work** The fact that the greedy algorithm was a constant-factor approximation algorithm for *min-sum set cover* was implicit in the work of Bay-Noy et al. [BNBH<sup>+</sup>98], and was made explicit in papers by Feige et al., who also simplified the proofs, both in the conference version [FLT02], and then further in the journal version [FLT04]. They also showed that the 4-approximation was the best possible unless  $P=NP$ . Other variants of this problem have been studied in different contexts, like when the set coverage is probabilistic (stochastic) [CFK03], or when the cost of a set depends on the set of uncovered elements at the time when it is picked [MBMW05].

At the other end of the spectrum is the *min-latency set cover* problem. This was formally studied by Hassin and Levin [HL05], who gave a factor  $e$ -approximation for the problem via techniques similar to those for the min-latency tour, a.k.a. the traveling repairman problem. Subsequently, they observed that min-latency set cover can be modeled as a special case of the classic precedence-constrained scheduling problem  $1|prec|\sum_j w_j C_j$ , for which several 2-approximation algorithms are known using a variety of different techniques (see, e.g., [CK04, KSW99] for surveys). This special case corresponds to the so-called “bipartite constraints” case, where there are two types of jobs  $J_1$  and  $J_2$ . All jobs in  $J_1$  have  $w_j = 0, p_j = 1$  (these correspond to elements), all jobs in  $J_2$  have  $w_j = 1, p_j = 0$  (these correspond to sets  $S_j \subset J_1$ ) and the precedence constraints have the form that each job  $j \in J_2$  must be preceded by the jobs  $S_j \subset J_1$ . To see the equivalence to min-latency set cover problem, note that any valid schedule is just an ordering of jobs in  $J_1$  (as jobs in  $J_2$  have size 0). Moreover, only jobs in  $J_2$  contribute to completion time (as jobs in  $J_1$  have weight 0), and being of size 0, a job in  $J_2$  can be assumed to be completed immediately after its preceding jobs in  $J_1$  have been scheduled. Woeginger [Woe03] showed that this special case (or equivalently the min-latency set cover problem) is as hard to approximate as the general  $1|prec|\sum_j w_j C_j$  problem. Recently it has been shown [BK09], that assuming a variant of the Unique Games Conjecture, it is hard to approximate  $1|prec|\sum_j w_j C_j$ , and hence min-latency set cover, to better than  $2 - \epsilon$  for any  $\epsilon > 0$ .

**Multiple Intents Re-Ranking:** The *multiple intents re-ranking* problem was introduced by Azar et al. [AGY09]. In this problem, each set  $S$  has a *weight* vector  $\overline{w}_S$  of length  $|S|$ , and if the elements of the set are output at times  $\overline{\tau}_S = (t_1, t_2, \dots, t_{|S|})$  where  $t_1 < t_2 < \dots < t_{|S|}$ , then the cost of the set is  $\overline{w}_S \cdot \overline{\tau}_S$ ; the goal is find an ordering of the elements that minimizes the sum of these costs  $\sum_{S \in \mathcal{S}} \overline{w}_S \cdot \overline{\tau}_S$ . (However, as noticed in that paper, by making copies of sets, one can equivalently imagine each set  $S$  to have a single requirement  $K(S)$ , and we are charged for the first time at which  $K(S)$  elements from  $S$  have been chosen; i.e., the model we use.) They showed that if all the weight vectors were increasing or decreasing, one could get constant factor approximations, even though the naïve greedy algorithm could be arbitrarily bad. They then gave an  $O(\log r)$ -approximation for a greedy-like algorithm using a clever harmonic interpolation idea; here  $r$  is the size of the largest set in the set system. However, we can show (see Section 6.1) that their algorithm cannot give a constant-factor approximation for the general problem.

## 2 Min-sum Set Cover and GenMSSC

A key difference between min-sum set cover and the generalized version of the problem can be illustrated by looking at the *max-coverage* variants of both these problems. In the max-coverage problem, given a bound  $k$ , the goal is to choose  $k$  elements which maximizes the number of sets hit. While it is known that the greedy algorithm is a  $1 - 1/e$  approximation algorithm for this problem, the max-coverage variant of the generalized problem becomes *Dense- $K$ -Subgraph* hard even for the case when a set is covered when 2 of its elements are selected. Indeed, given a graph  $G$ , consider the following instance of *GenMSSC*: elements are the vertices, and sets the edges. Each set  $e = \{u, v\}$  has a covering requirement  $K(e) = 2$ . Clearly, the set of  $k$  elements/vertices which “hits” the most number of sets/edges is the collection of  $k$  vertices which induces the most number of edges. Therefore, the *max-coverage* version of *GenMSSC* is as hard as the *Dense- $K$ -Subgraph* problem.

Hence, while one can get constant factor approximations for the min-sum set cover problem by solving the max-coverage problem for bounds of  $2^i$  (for  $1 \leq i \leq \lceil \log n \rceil$ ) and combining these solutions to get a global linear ordering, naïvely using this approach would fail for the *GenMSSC* problem. (Hassin and Levin [HL05] use the max-coverage approach differently for their  $e$ -approximation, and it would be interesting to see if that approach can be extended to work for *GenMSSC*.)

Our approach is based on a variation of this idea.

In particular, we use the following observation, which suffices for our purposes even though it is too weak to yield a useful guarantee for max-coverage. Consider the LP formulation for the max-coverage instance given a bound  $k$ , strengthened by adding the knapsack cover inequalities. Let  $\ell$  denote the number of sets which are covered fractionally to an extent of at least  $1/2$  (or any constant) in an optimal fractional solution. Then the solution obtained by applying a round of randomized rounding (to the LP solution scaled by a suitable constant factor) covers at least  $\Omega(\ell)$  sets. At a high level, it is this observation that forms the basis of our algorithm and its analysis. We next describe the details.

### 3 An LP Relaxation

Let  $[n] = \{1, 2, \dots, n\}$ , where  $n = |U|$ , the number of elements in the universe. In the LP relaxation given in Figure 3,  $x_{et}$  is the indicator variable for whether element  $e \in U$  is selected at time  $t \in [n]$ , and  $y_{St}$  is the indicator variable for whether set  $S$  has been covered before time  $t \in [n]$ .

If  $x_{et}$  and  $y_{St}$  are restricted to only take values 0 or 1, then this is easily seen to be a valid formulation for the problem. In particular, Constraints (3.1) require that only one element can be assigned to a time slot and constraints (3.2) require that each element must be assigned some time slot. Constraints (3.3) correspond to the *knapsack cover constraints* and require that if  $y_{St} = 1$ , then for every subset of elements  $A$ , at least  $K(S) - |A|$  elements must be chosen from the set  $S \setminus A$  before time  $t$ . As a consequence, we get that  $y_{St}$  can be 1 if and only if there have been  $K(S)$  elements picked from  $S$  before time  $t$ . Therefore, the set would incur an LP cost of exactly the cover time of the set in the integral ordering (since the term  $(1 - y_{St})$  would keep contributing 1 to the LP objective until the set has been covered).

Let  $\text{Opt}$  denote any optimal solution of the given GenMSSC instance, and let  $\text{LPOpt}$  denote the cost of an optimal LP solution. From the above discussion, the LP is a valid relaxation and hence we have that,

**LEMMA 3.1.** *The LP cost  $\text{LPOpt}$  is at most the total cover time of an optimal solution  $\text{Opt}$ .*

#### 3.1 Solving the LP: The Separation Oracle

Even though the LP formulation has an exponential number of constraints, it can be solved assuming we can, in polynomial time, verify if a candidate solution  $(x, y)$  satisfies all the constraints. Indeed, consider any fractional solution  $(x, y)$ . Constraints (3.1), (3.2), and (3.4) can easily be verified in  $O(mn + n^2)$  time, one by one.

Consider any set  $S$ , a time instant  $t$  and a particular size  $a < K(S)$ . To verify constraint (3.3), we wish to check the following condition:

$$(3.5) \quad \min_{A: |A|=a} \sum_{e \in S \setminus A} \sum_{t' < t} x_{et'} \geq (K(S) - a) \cdot y_{St}$$

Now, notice that for any fixed set  $A$  of size  $a$ , the left hand side could be rewritten as  $\sum_{e \in S} \sum_{t' < t} x_{et'} - \sum_{e \in A} \sum_{t' < t} x_{et'}$ . Therefore, if the above condition were to hold when we choose  $A$  to be the set of the  $a$  elements with the *largest values* of  $\sum_{t' < t} x_{et'}$ , then it would also hold for any other set  $A$ . Hence we can verify constraint (3.3) in polynomial time for each choice of set  $S$ , time  $t$ , and size  $a$ , and there are only  $O(mn^2)$  such choices to iterate over.

### 4 The Rounding Algorithm

Let  $(x^*, y^*)$  denote the optimal LP solution. Our rounding algorithm proceeds in  $O(\log n)$  stages, with the  $i^{\text{th}}$  stage operating in the time interval  $[1, 2^i)$ . In stage  $i$ , we perform one round of randomized rounding (as described below) on the fractional solution restricted to the interval  $[1, 2^i)$  and obtain a set  $O_i$  of elements. At the conclusion of these stages, we output the elements of  $O_1$ , followed by elements of  $O_2, O_3, \dots, O_{\lceil \log n \rceil}$ , with the elements of any set  $O_j$  being output in an arbitrary order. (Of course, we should only keep the first occurrence of any element in the final output, but imagining elements to potentially be output multiple times will be easier for the analysis.)

The rounding process for stage  $i$  that generates the set  $O_i$  is the following:

---

#### Algorithm 1 Randomized Rounding for stage $i$

---

- 1: **let**  $t_i = 2^i$ .
  - 2: **let**  $z_{e,i} \leftarrow \sum_{t' < t_i} x_{et'}^*$  be the fractional extent to which  $e$  is selected before time  $t_i$ , for each  $e \in U$ .
  - 3: **let**  $p_{e,i} \leftarrow \min(1, 8z_{e,i})$  for all  $e \in U$ .
  - 4: **mark** each element  $e \in U$  independently with probability  $p_{e,i}$ .
  - 5: **let**  $O_i$  be the set of marked elements.
  - 6: **if**  $|O_i| > 16 \cdot 2^i$  **then** drop all but  $16 \cdot 2^i$  elements from  $O_i$ .
  - 7: **return**  $O_i$ .
- 

### 5 The Analysis

In the interests of expositional simplicity, we have not tried to optimize the constants in our analysis.

**OBSERVATION 5.1.** *The fractional coverage of the sets is monotonically non-decreasing. That is,  $y_{St}^* \geq y_{St'}^*$  for all sets  $S \in \mathcal{S}$  and  $1 \leq t' \leq t \leq n$ .*

$$\begin{aligned}
(1P) \quad & \text{minimize} \quad \sum_{1 \leq t \leq |U|} \sum_{S \in \mathcal{S}} (1 - y_{St}) \\
(3.1) \quad & \text{subject to} \quad \sum_{e \in U} x_{et} = 1 \quad \forall t \in [n] \\
(3.2) \quad & \sum_{t \in [n]} x_{et} = 1 \quad \forall e \in U \\
(3.3) \quad & \sum_{e \in S \setminus A} \sum_{t' < t} x_{et'} \geq (K(S) - |A|) \cdot y_{St} \quad \forall S \in \mathcal{S}, \forall A \subseteq S, \forall t \in [n] \\
(3.4) \quad & x_{et}, y_{St} \in [0, 1] \quad \forall e \in U, S \in \mathcal{S}, t \in [n]
\end{aligned}$$

Figure 3.1: A Linear Programming Relaxation for GenMSSC

For any set  $S \in \mathcal{S}$ , let  $t_S^*$  denote the last time  $t$  at which  $y_{St}^* \leq 1/2$ . The following fact is then immediate because the set  $S$  pays at least  $(1 - y_{St}^*) \geq 1/2$  for each time  $t \in [1, t_S^*]$ .

FACT 5.1. (LOWER BOUND)  $\text{LPOpt} \geq \frac{1}{2} \sum_S t_S^*$

LEMMA 5.1. *For any set  $S$  and any stage  $i$  such that  $t_S^* \in [1, t_i]$ , the probability that  $K(S)$  elements from  $S$  are not marked in stage  $i$  is at most  $e^{-9/8}$ .*

**Proof.** Consider any set  $S$ , and let  $S_g = \{e \in S \mid z_{e,i} \geq 1/8\}$ . By the choice of  $p_{e,i}$  in step 3 of our rounding procedure, we know that all elements in  $S_g$  are definitely marked in stage  $i$ , and any element  $e \in S \setminus S_g$  is independently marked with probability  $8z_{e,i}$ . Thus, if  $|S_g| \geq K(S)$ , then clearly the lemma holds.

Thus we consider the case when  $|S_g| < K(S)$ . Recall that we are considering a set  $S$  and stage  $i$  such that  $t_S^* \in [1, t_i]$ ; since  $t_S^*$  was the last time  $t$  at which  $y_{St}^* \leq \frac{1}{2}$ , it follows that  $y_{St_i}^* > \frac{1}{2}$ . Hence, setting  $A = S_g$ , constraint (3.3) implies that

$$\begin{aligned}
\sum_{e \in S \setminus S_g} z_{e,i} &= \sum_{e \in S \setminus S_g} \sum_{t' < t_i} x_{et'}^* \geq (K(S) - |S_g|) \cdot y_{St_i}^* \\
&\geq \frac{1}{2} (K(S) - |S_g|).
\end{aligned}$$

Therefore, the expected number of elements from  $S \setminus S_g$  marked in stage  $i$  is

$$\sum_{e \in S \setminus S_g} 8z_{e,i} \geq 4(K(S) - |S_g|)$$

Since these elements are marked independently of each other, we can use the following Chernoff bound [MR95] (Theorem 4.2): if  $X_1, X_2, \dots, X_n$  are independent  $\{0, 1\}$ -valued random variables with  $X = \sum_i X_i$  such

that  $\mathbb{E}[X] = \mu$ , then

$$\Pr[X < \mu(1 - \beta)] \leq \exp(-\frac{\beta^2}{2}\mu)$$

For our application, since we have  $\mu \geq 4(K(S) - |S_g|) \geq 4$ , we can substitute  $\beta = \frac{3}{4}$  and bound the tail probability that fewer than  $(K(S) - |S_g|)$  elements are marked from  $S \setminus S_g$  by  $\exp(-\frac{(3/4)^2}{2} \cdot 4) = e^{-9/8}$ . As the elements in  $S_g$  are all marked with probability 1, it follows that the probability that fewer than  $K(S)$  elements are marked from  $S$  is also at most  $e^{-9/8}$ . ■

LEMMA 5.2. *The probability that any elements are dropped in step 6 is at most  $e^{-6}$ .*

**Proof.** To show this, we use the following concentration inequality [BLM00] (Theorem 1, Remark 3): if  $X_1, X_2, \dots, X_n$  are independent  $\{0, 1\}$ -valued random variables with  $X = \sum_i X_i$  such that  $\mathbb{E}[X] = \mu$ , then

$$\Pr[X > \mu + \beta] \leq \exp(-\frac{\beta^2}{2\mu + 2\beta/3})$$

In our setting, since the probability with which an element is picked in  $O_i$  is at most 8 times the extent to which it was scheduled in  $[1, 2^i]$  by the fractional LP solution, the expected number of elements picked (i.e.  $\mu$ ) in  $O_i$  is at most  $8 \cdot 2^i$ . Therefore, by substituting  $\beta = 8 \cdot 2^i$  and  $\mu \leq 8 \cdot 2^i$  in the above inequality, we get that the probability of picking more than  $16 \cdot 2^i$  is at most  $\exp(\frac{-64 \cdot 2^{2i}}{(64/3)2^i}) \leq \exp(-6)$ . ■

We now bound the cover time of the set  $S$  for the above algorithm.

THEOREM 5.1. (COVER TIME) *The expected cover time of a set  $S$  is at most  $O(1)t_S^*$ .*



**Proof.** Let  $\text{Cov}^{\text{Alg}}(S)$  denote the cover time of set  $S$  with respect to the ordering output by our algorithm. For ease of analysis, we will consider a set  $S$  to be covered in some stage  $i$  only if that  $t_S^* \in [1, t_i)$ , and moreover the set  $O_i$  returned is not truncated. Note that if the set  $S$  is actually covered with any of these criteria not met, its cover time only improves. Let  $\mathcal{E}_{iS}$  denote the event that set  $S$  is first covered in stage  $i$  under this modified notion of coverage. Then we have

$$\mathbb{E} [\text{Cov}^{\text{Alg}}(S)] \leq \sum_{i=\lceil \log t_S^* \rceil}^{\lceil \log n \rceil + 1} (2 \cdot 16 \cdot 2^i) \times \Pr [\mathcal{E}_{iS}],$$

since if  $S$  is covered in stage  $i$ , its cover time is at most  $\sum_{j=1}^i 16 \cdot 2^j \leq 32 \cdot 2^i$ . Also, we know that any set will certainly be covered by stage  $\lceil \log n \rceil$  because the matching constraints (3.1) and (3.2) would ensure that each element be picked to an extent 1 by time  $n$ .

Now, the event  $\mathcal{E}_{iS}$  that a set  $S$  is *first* covered in stage  $i$  is strictly contained in the event that  $S$  is *not covered* in stages  $\lceil \log t_S^* \rceil, (\lceil \log t_S^* \rceil + 1), \dots, (i-2)$ , and  $(i-1)$ . But for any  $i$ , the event that  $S$  is not covered in stage  $i$  occurs only when either

1.  $K(S)$  elements from  $S$  were not picked in  $O_i$ , or
2.  $O_i$  was truncated in step 6.

The former event happens with probability at most  $e^{-9/8}$  from Lemma 5.1, and the latter event happens with probability at most  $e^{-6}$  from Lemma 5.2. Hence, the probability that  $S$  is not covered in any fixed stage is at most  $e^{-9/8} + e^{-6} < e^{-1}$ . Thus, we have

$$\Pr [\mathcal{E}_{iS}] \leq \exp(-(i - \lceil \log t_S^* \rceil)).$$

Plugging this into equation (5.6), we get

$$\begin{aligned} \mathbb{E} [\text{Cov}^{\text{Alg}}(S)] &\leq \sum_{i=\lceil \log t_S^* \rceil}^{\lceil \log n \rceil} (2 \cdot 16 \cdot 2^i) \times e^{-(i - \lceil \log t_S^* \rceil)} \\ &= 32 \cdot 2^{\lceil \log t_S^* \rceil} \times \sum_{i=\lceil \log t_S^* \rceil}^{\lceil \log n \rceil} (2/e)^{(i - \lceil \log t_S^* \rceil)} \\ &\leq (64 \cdot \frac{e}{e-2}) \times t_S^* \end{aligned}$$

By linearity of expectation, the expected covering time of all the sets is at most  $\frac{64e}{e-2} \sum_S t_S^*$ , which by Fact 5.1 is at most  $\frac{128e}{e-2} \text{LPOpt} \leq 485 \text{LPOpt}$ . This completes the proof of Theorem 1.1. ■

## 6 Two Examples

**6.1 A Bad Example for Harmonic Interpolation-Based Greedy** We now give an example where the algorithm of [AGY09] has an approximation ratio of  $\Omega(\sqrt{\log n})$  for the *multiple intents re-ranking* problem. (See Section 1.1 for the problem definition; recall that it is equivalent to our problem.)

Consider the following set system: the universe is  $U = \{a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_t\}$ . There is a “large” set  $S_0 = \{a_1, a_2, \dots, a_n\}$  with a weight vector  $\bar{w}_{S_0} = (1, 1, \dots, 1)$ . There are also  $t$  other “small” sets  $S_1, S_2, \dots, S_t$ , with the set  $S_i = \{b_i\}$  having a weight vector  $\bar{w}_{S_i} = (H_n/2)$ . Here,  $H_n$  is the  $n^{\text{th}}$  harmonic number, defined by  $H_n = 1 + 1/2 + \dots + 1/n$ .

Consider the ordering  $b_1, b_2, \dots, b_t, a_1, a_2, \dots, a_n$  of the vertices (henceforth called *order A*): . In this ordering, any small set  $S_i$  would incur a (weighted) cost of exactly  $iH_n/2$ , while the large set incurs a cost of  $(t+1) + (t+2) + \dots + (t+n) = nt + \Theta(n^2)$ . Therefore, the total weighted cost of such an order would be  $O(t^2 H_n/2 + nt + n^2)$ .

The harmonic interpolation method [AGY09] was to replace each weight vector  $\bar{w} = (w_1, w_2, \dots, w_l)$  by a new “harmonic” weight vector  $\bar{w}' = (w'_1, \dots, w'_l)$ , where  $w'_j = \sum_{j' \geq j} w_{j'} \cdot \frac{1}{j' - j + 1}$ , and then run the greedy algorithm on these new weight vectors. Note that the harmonic weight vector for the small sets remains the same as the original weight vector, but the one for the large set changes to  $\bar{w}'_{S_0} = (H_n, H_{n-1}, \dots, H_{n/2}, \dots, 1)$ . Now the greedy algorithm would not pick any of the vertices from  $\{b_i : i \in [1, t]\}$  during the first  $n/2$  time instants, since the weight vector for  $S_0$  has larger values. Therefore, each of the small sets would incur a cost of at least  $\frac{n}{2} H_n/2$ , and the large set incurs a cost of  $\Omega(n^2)$ . As a result, the total cost for the instance under this ordering would be  $\Omega(nt H_n/2 + n^2)$ .

Setting  $t = n(\log n)^{-1/2}$ , we see that the cost incurred by order A is  $O(n^2)$  whilst the cost incurred by the harmonic algorithm is  $\Omega(n^2 \cdot \sqrt{\log n})$ , and this gives us an algorithmic gap of  $\Omega(\sqrt{\log n})$ .

**6.2 A Bad Example for the Standard LP Relaxation** Consider the LP relaxation without the knapsack cover inequalities for the GenMSSC problem.

We now show that the integrality gap of this LP can be arbitrarily bad for large values of  $K(S)$ . Consider the following universe:  $U = l\{a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_l\}$ . There are  $l$  sets, with set  $S_i = \{a_1, a_2, \dots, a_n, b_i\}$  for each  $i \in [l]$ . Moreover, all the sets have a covering requirement of  $K(S) = (n+1)$ . Note that this is just an instance of the min-latency set cover problem, since any set is covered only when all the elements contained in it are selected.

$$\begin{aligned}
(\text{LP}_{\text{noKC}}) \quad & \text{minimize} \quad \sum_{1 \leq t \leq |U|} \sum_{S \in \mathcal{S}} (1 - y_{St}) \\
(6.6) \quad & \text{subject to} \quad \sum_{e \in U} x_{et} \leq 1 \quad \forall t \in [n] \\
(6.7) \quad & \sum_{t \in [n]} x_{et} \leq 1 \quad \forall e \in U \\
(6.8) \quad & \sum_{e \in S} \sum_{t' < t} x_{et'} \geq K(S) \cdot y_{St} \quad \forall S \in \mathcal{S}, \forall t \in [n] \\
(6.9) \quad & x_{et}, y_{St} \in [0, 1] \quad \forall e \in U, S \in \mathcal{S}, t \in [n]
\end{aligned}$$

Figure 6.2: The Natural LP Relaxation for GenMSSC

Consider the following fractional solution: Select element  $a_t$  in time slot  $t$  for  $t \in [n]$ , and element  $b_t$  in time slot  $(n + t)$  for  $t \in [l]$  (i.e. set  $x_{a_t t} = 1$  and  $x_{b_t(n+t)} = 1$ ). Although the  $x_{et}$  variables are integral, the LP will now cheat when it comes to the  $y_{St}$  variables (which is what contributes to the objective). For any set  $S$  and time slot  $t$ , the LP solution sets  $y_{St} = \min(1, \frac{1}{n+1} \sum_{e \in S} \sum_{t' < t} x_{et'})$ .

We first analyze the LP cost of this assignment: clearly, with each element we pick from  $\{a_1, a_2, \dots, a_n\}$ , the  $y_{St}$  term would decrease by an additive  $1/(n+1)$ ; and this happens for each set, in each of the first  $n$  time steps. Hence, each set would incur a cost of  $1 + (1 - \frac{1}{n+1}) + (1 - \frac{2}{n+1}) + \dots + (1 - \frac{n-1}{n+1})$  (roughly  $n/2$ ) for the first  $n$  time-steps. After this, however, we cover one set at a time in time slots  $(n+1), (n+2), \dots, (n+l)$ : but by this time each set has only a  $1/(n+1)$  uncovered fraction which needs to incur any cost for this final covering step: hence the total LP cost would be  $\Theta(nl + \frac{1}{n+1}(nl + l^2))$ .

However, each integer solution might as well schedule the elements  $\{a_1, a_2, \dots, a_n\}$  before selecting the elements  $\{b_1, b_2, \dots, b_l\}$  one by one, giving a cost of  $(n+1) + (n+2) + \dots + (n+l) = nl + l^2$ . Now setting  $n = \sqrt{l}$  gives us an integrality gap of  $\Omega(\sqrt{l})$ .

Notice that with the knapsack cover inequality, the  $y_{St}$  values cannot decrease by  $1/(n+1)$  with each additional covered element. In fact, for this extreme case of min-latency, the above LP strengthened with the knapsack cover inequalities is equivalent to the time-indexed LP relaxation for precedence-constrained scheduling on a single machine, which has an integrality gap of 2.

## 7 Closing Remarks

The proofs trivially extend to the case where each set  $S$  also has a weight  $w_S \in \mathbb{R}_+$ , and the objective function is  $\sum_S w_S \text{Cov}(S)$ .

The current approximation factor is a rather large

constant, and it would be interesting to pin down the integrality gap of this LP relaxation better. For both the extreme cases of the min-sum set cover and min-latency set-cover, it is known that the integrality gap of this LP relaxation is 4 (see [FLT02] for a dual-fitting proof) and 2 [HSSW97] respectively. We have not tried to optimize the constants in this abstract; however, getting a substantially lower constant might require other ideas.

## References

- [AGY09] Yossi Azar, Iftah Gamzu, and Xiaoxin Yin. Multiple intents re-ranking. In *STOC '09: Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, pages 669–678, New York, NY, USA, 2009. ACM.
- [BK09] Nikhil Bansal and Subhash Khot. Optimal long code test with one free bit. In *FOCS*, page to appear, 2009.
- [BLM00] Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. A sharp concentration inequality with application. *Random Struct. Algorithms*, 16(3):277–292, 2000.
- [BNBH<sup>+</sup>98] Amotz Bar-Noy, Mihir Bellare, Magnús M. Halldórsson, Hadas Shachnai, and Tami Tamir. On chromatic sums and distributed resource allocation. *Inform. and Comput.*, 140(2):183–202, 1998.
- [CFK03] Edith Cohen, Amos Fiat, and Haim Kaplan. Efficient sequences of trials. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (Baltimore, MD, 2003)*, pages 737–746, New York, 2003. ACM.
- [CFLP00] Robert Carr, Lisa Fleischer, Vitus Leung, and Cynthia Phillips. Strengthening integrality gaps for capacitated network design and covering problems. In *Symposium on Discrete Algorithms (SODA)*, pages 106–115, 2000.
- [CK04] Chandra Chekuri and Sanjeev Khanna. Approximation algorithms for minimizing average weighted

- completion time. In Joseph Leung, Laurie Kelly, and James H. Anderson, editors, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*. CRC Press, Inc., Boca Raton, FL, USA, 2004.
- [CM99] Chandra Chekuri and Rajeev Motwani. Precedence constrained scheduling to minimize sum of weighted completion times on a single machine. *Discrete Appl. Math.*, 98(1-2):29–38, 1999.
- [FLT02] Uriel Feige, László Lovász, and Prasad Tetali. Approximating min-sum set cover. In *Approximation algorithms for combinatorial optimization*, volume 2462 of *Lecture Notes in Comput. Sci.*, pages 94–107. Springer, Berlin, 2002.
- [FLT04] Uriel Feige, László Lovász, and Prasad Tetali. Approximating min sum set cover. *Algorithmica*, 40(4):219–234, 2004.
- [HL05] Refael Hassin and Asaf Levin. An approximation algorithm for the minimum latency set cover problem. In *Algorithms—ESA 2005*, volume 3669 of *Lecture Notes in Comput. Sci.*, pages 726–733. Springer, Berlin, 2005.
- [HSSW97] Leslie A. Hall, Andreas S. Schulz, David B. Shmoys, and Joel Wein. Scheduling to minimize average completion time: off-line and on-line approximation algorithms. *Math. Oper. Res.*, 22(3):513–544, 1997.
- [KSW99] David Karger, Cliff Stein, and Joel Wein. Scheduling algorithms. In *Algorithms and theory of computation handbook*, pages 35–1–35–33. CRC, Boca Raton, FL, 1999.
- [MBMW05] Kamesh Munagala, Shivnath Babu, Rajeev Motwani, and Jennifer Widom. The pipelined set cover problem. In *Database theory—ICDT 2005*, volume 3363 of *Lecture Notes in Comput. Sci.*, pages 83–98. Springer, Berlin, 2005.
- [MQW03] François Margot, Maurice Queyranne, and Yaoguang Wang. Decompositions, network flows, and a precedence constrained single-machine scheduling problem. *Oper. Res.*, 51(6):981–992, 2003.
- [MR95] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.
- [Woe03] Gerhard Woeginger. On the approximability of average completion time scheduling under precedence constraints. *Discrete Applied Mathematics*, 131:237–252, 2003.