Journal of Scheduling **8:** 97–106, 2005.

SCHEDULING WEB ADVERTISEMENTS: A NOTE ON THE MINSPACE PROBLEM

MILIND DAWANDE[1], SUBODHA KUMAR[2] AND CHELLIAH SRISKANDARAJAH[1]

[1] *University of Texas at Dallas, Richardson, TX 75080, USA*
[2] *University of Washington, Business School, Seattle, WA 98195, USA*

ABSTRACT

Free services to internet users are commonly available on many web sites e.g., Hotmail and Yahoo. For such sites, revenue generated from advertisements (hereafter also called "ads") placed on the web pages is critical for survival. An effective way to schedule ads on web pages to optimize certain performance measures is an important problem that these sites need to address.

In this note, we report improved approximation algorithms for the following problem: ads from a set of n ads $A = \{A_i, \ldots, A_n\}$ are to be placed on a web page during a planning horizon that is divided into N time intervals. In each time interval, ads are shown in a rectangular space called a *slot*. An ad A_i is specified by its size s_i and frequency w_i and is to be scheduled in exactly w_i slots. We are required to find a schedule that minimizes the maximum fullness among all the slots, where the fullness of a slot is the sum of the sizes of ads scheduled in that slot. Our results include (i) the first online algorithm with a performance bound of $2 - \frac{1}{N}$, and (ii) two offline algorithms with performance guarantees of $1 + 1\frac{1}{\sqrt{2}}$ and $\frac{3}{2}$, respectively. These bounds are significant improvements over those for previously known algorithms presented in Adler, Gibbons, and Matias (2002) and Dawande, Kumar, and Sriskandarajah (2003).

KEY WORDS: web advertising, scheduling, approximation algorithms

1. INTRODUCTION

The popularity of the world wide web has increased dramatically in the last few years. At the end of 1999, the number of web users was nearly 260 million worldwide, whereas this number is estimated to be around 765 million by the end of 2005—thus tripling in six years (Puetz, 2000). This increasing popularity of the web has made it an attractive medium for advertisers. To quote a finding from a Millward Brown study (Hyland, 2000): "Single exposure to a web banner generated greater awareness than a single exposure to a television or print ad". Since the ultimate goal of advertisers is to reach customers effectively, the market for web advertising is growing significantly.

The Interactive Advertising Bureau (http://www.iab.net) has reported significant web advertising revenues in recent years: \$4.6 billion in 1999, \$8 billion in 2000, \$7 billion in 2001 and \$6 billion in 2002.

Many web sites (e.g., Hotmail and Yahoo) have adopted business models in which a considerable fraction of the revenue is generated from advertisements displayed on web pages. For such sites, it is very important to maximize the revenue from advertising without significantly sacrificing the non-advertisement content. Although one factor considered in setting advertising charges is the number of hits on the site, others that come into play are the size of the ad, and how often it is displayed

Correspondence to: Subodha Kumar. E-mail: subodha@u.washington.edu

on a page. Two models for scheduling web advertisements, namely the MINSPACE problem and MAXSPACE problem, have been recently proposed (Adler, Gibbons, and Matias, 2002). Both problems are NP-hard in the strong sense (Adler, Gibbons, and Matias, 2002) and, therefore, polynomial-time approximation algorithms for the problems have been reported in the literature. For the MAXSPACE problem, studied in Adler, Gibbons, and Matias (2002), Dawande, Kumar, and Sriskandarajah (2003) and Fruend and Naor (2003), the best known result is a $\frac{1}{3}$-approximation reported in Fruend and Naor (2003). For the MINSPACE problem (Adler, Gibbons, and Matias, 2002; Dawande, Kumar, and Sriskandarajah, 2003), the best known result is a 2-approximation reported in Adler, Gibbons, and Matias (2002) and Dawande, Kumar, and Sriskandarajah (2003). The purpose of this note is to report three results for the MINSPACE problem: (a) an offline $(1 + \frac{1}{\sqrt{2}})$-approximation algorithm, (b) an offline $\frac{3}{2}$-approximation algorithm, and (c) an online $(2 - \frac{1}{N})$-approximation algorithm.

This note is organized as follows. Section 2 defines the MINSPACE problem. In Section 3, we describe the algorithms and analyze their performance. Conclusions are provided in Section 4.

2. PROBLEM DESCRIPTION

Consider a set of n ads $A = \{A_1, \ldots, A_n\}$. A planning horizon is divided into N equal time intervals. In each time interval, the selected ads are displayed in a rectangular space called a *slot*. A slot usually appears on the left, right, top or bottom of the web page and houses one or more ads. An ad placed in a particular slot will be displayed (to users visiting the web site) for the time interval corresponding to that slot. Each ad has two characteristics: *size* and *frequency*. The size of an ad A_i is denoted by s_i and represents the amount of space A_i occupies in a slot. The frequency of an ad A_i is denoted by w_i and is the number of slots which should contain a copy of the ad. Advertisers do not want to display more than one copy of an ad in the same slot and therefore an ad A_i can be displayed at most once in a slot.

Definition. Ad A_i is said to be *scheduled* if a total of w_i copies of A_i appear in the slots with at most one copy in a slot.

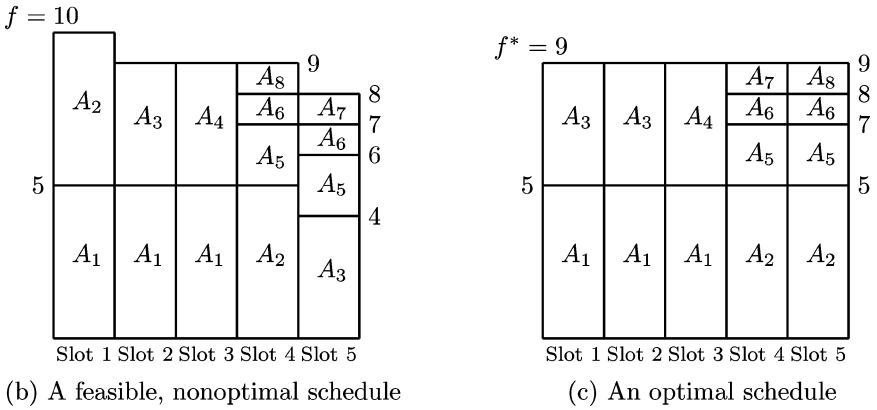
Clearly, $w_i \leq N, \forall i$. We assume that the width of the ads placed in a slot is same as the width of the slot. The *fullness* of a slot j is defined as $f_j = \sum_{i:A_i \in B_j} s_i$, where B_j is a set of ads which have a copy in slot j. The *height* of the schedule, f, is the maximum fullness of the slots, i.e., $f = \max_j f_j$.

In the MINSPACE problem, all the ads are to be placed in N slots. The objective is to find a schedule with minimum height, say f^*. For example, consider the problem instance given in Figure 1(a). A feasible schedule is shown in Figure 1(b). Here, the fullness of slot 2 is $s_1 + s_3 = 9$ and the height of the schedule is $f = 10$. Figure 1(c) shows an optimal schedule with $f^* = 9$. Note that during any time interval only one of these five slots is shown to a user accessing the web page. We assume that an ad can be scheduled in any slot and do not consider the problem of deciding which slot to display in a particular time interval.

In the special case when $w_i = 1, \forall i$, this problem is equivalent to the classical multiprocessor scheduling problem (MSP): given n independent tasks, each of which requires processing on any of m identical parallel processors, find a schedule that minimizes the total completion time of all tasks (or the *makespan* of the schedule). Here, the number of slots, N, is equivalent to the m identical parallel processors while the size s_i of an ad A_i corresponds to the processing time of a task. The LPT (largest processing time first) algorithm, with a worst-case bound of $\frac{4}{3} - \frac{1}{3m}$, is a

Ad A_i	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
s_i	5	5	4	4	2	1	1	1
w_i	3	2	2	1	2	2	1	1

(a) Problem data



(b) A feasible, nonoptimal schedule (c) An optimal schedule

Figure 1. An example to illustrate the MINSPACE problem.

well-known algorithm for MSP (Graham, 1969). Other algorithms for MSP include (i) the multifit algorithm (Coffman, Garey, and Johnson, 1978; Friesen, 1984) with a performance guarantee of $\frac{6}{5}$, (ii) an improved multifit algorithm (Friesen and Langston, 1986) with a guarantee of $\frac{72}{61}$ and (iii) a PTAS (Hochbaum and Shmoys, 1987). Several approximations algorithms for the online version of the MSP are available in the literature (Graham, 1969; Galambos and Woeginger, 1993; Chandrasekaran et al., 1997; Bartal et al., 1995; Karger, Phillips, and Torng, 1996; Albers, 1999; Fleischer and Wahl, 2000).

3. THE MINSPACE PROBLEM

We first provide an integer programming formulation for the MINSPACE problem (corresponding to an ad set $A = \{A_1, \ldots, A_n\}$).

$$\begin{aligned} \min \quad & F \\ \text{subject to} \quad & F \geq \sum_{i=1}^{n} s_i x_{ij} \quad j = 1, 2, \ldots, N \qquad (IP_{\min}(A)) \\ & \sum_{j=1}^{N} x_{ij} = w_i \quad i = 1, 2, \ldots, n \end{aligned}$$

where

$$\begin{aligned} x_{ij} &= 1 \quad \text{if ad } A_i \text{ is scheduled in slot } j, \\ &= 0 \quad \text{otherwise.} \end{aligned}$$

Note: For simplicity of notation, we will use $IP_{\min}$ and $LP_{\min}$ to denote $IP_{\min}(A)$ and its linear programming relaxation $LP_{\min}(A)$, respectively. However, if a MINSPACE problem corresponds to a set of ads $U \subset A$, we will explicitly use $IP_{\min}(U)$ and $LP_{\min}(U)$.

The objective function and first set of constraints minimize the height of the schedule. The feasibility is ensured by the second set of constraints, which guarantee that for an ad A_i, a total of w_i copies are scheduled in the slots. Since $x_{ij} \in \{0, 1\}$, at most one copy of an ad A_i can be scheduled in any slot j. Since the MINSPACE problem is NP-Hard (Adler, Gibbons, and Matias, 2002), our focus in this paper is on designing polynomial time heuristics which provide solutions with a performance guarantee. In particular, we are interested in heuristics which provide a solution within a *constant* factor of the optimum solution.

Our results include:

1. An online $(2 - \frac{1}{N})$-approximation algorithm.
2. A (offline) $(1 + \frac{1}{\sqrt{2}})$-approximation algorithm.
3. A (offline) $\frac{3}{2}$-approximation algorithm.

Recall that, for a given schedule, f denotes the maximum slot fullness among all the slots, i.e., $f = \max_{1 \le j \le N} f_j$. The objective is to find a schedule with $f^* = \min f$, where the minimum is taken over all valid schedules. Clearly, f^* is the optimum solution value for $IP_{\min}$.

3.1. An online algorithm

In an online algorithm, decisions are made without prior knowledge about the ads arriving in future. We propose an online algorithm called First Come Least Full (FCLF). FCLF schedules a new ad to the least full slots. The algorithm uses the following rule:

Algorithm FCLF

Schedule a new ad A_i with size s_i and frequency w_i, to the w_i least full slots.

Theorem 1. The performance bound of FCLF is $\frac{f}{f^*} \le 2 - \frac{1}{N}$. *This bound is tight.*

Proof. After scheduling all the ads, let p and q be the slots with maximum and minimum fullness respectively. Let the fullness of slots p and q be F_p and F_q respectively. Therefore, the heuristic solution $f = F_p$. Without loss of generality, re-index the ads assigned in slot p as $A_1, A_2, A_3, \ldots$ such that ad A_1 is the last assigned ad, A_2 is the second last assigned ad and so on. If all ads in slot p are also in slot q then $F_p = F_q = \frac{\sum_{i=1}^{n} s_i w_i}{N}$ and the heuristic solution is optimal since $\frac{\sum_{i=1}^{n} s_i w_i}{N}$ is a lower bound on f^*. Otherwise, let A_k be the first ad in slot p which is not in slot q. Let s_k be the size of ad A_k.

Let $d = \sum_{i=1}^{k-1} s_i$. Just prior to scheduling ad A_k, let the fullness of slot p be c_p and the fullness of slot q be c_q. Since the ads are scheduled in the least full slots and A_k is assigned to slot p but not to slot q, we have $c_q \ge c_p$. Note that $F_q \ge d + c_q$ and $F_p = d + c_p + s_k$. Therefore, $F_q \ge F_p - s_k - c_p + c_q$ and hence

$$F_q \ge F_p - s_k \tag{1}$$

Table 1. An example to illustrate the tightness of the bound for algorithm FCLF

Ad A_i	A_1	A_2	A_3	A_4	.....	A_{N-1}	A_N	A_{N+1}
s_i	$K - \frac{K}{N}$	$\frac{K}{N}$	$\frac{K}{N}$	$\frac{K}{N}$		$\frac{K}{N}$	$\frac{K}{N}$	K
w_i	$N-1$	1	1	1		1	1	1

We note two simple lower bounds on the optimal solution f^*:

$$\begin{aligned} f^* &\geq \frac{\sum_{i=1}^{n} s_i w_i}{N} \\ &\geq \frac{NF_q + (F_p - F_q)}{N} \\ &= F_q + \frac{F_p - F_q}{N} \end{aligned} \tag{2}$$

and

$$f^* \geq s_k \tag{3}$$

Combining (1) and (2), we get $f^* \geq F_p(1 - \frac{1}{N}) - s_k(1 - \frac{1}{N}) + F_p(\frac{1}{N})$. Combining this with (3) gives $\frac{f}{f^*} \leq 2 - \frac{1}{N}$, because $f = F_p$.

To prove the tightness of the bound, consider an example with $N+1$ ads $A_i, i = 1, \ldots, N+1$, where N is the number of slots. The sizes and frequencies of the ads are given in Table 1 where K is any positive number. The ads arrive in the order of their indices. Algorithm FCLF assigns $N-1$ copies of ad A_1 to slots $1, 2, \ldots, N-1$. It then assigns ads A_j, $j = 2, 3, \ldots, N$ to slot N. Finally, it assigns ad A_{N+1} to slot 1. Therefore, $f = F_p =$ fullness of slot $1 = 2K - \frac{K}{N}$. In an optimal solution, ads A_1 and A_{j+1} are assigned to slot j, $j = 1, 2, \ldots, N-1$ and ad A_{N+1} is assigned to slot N. Therefore, $f^* = K$ and $\frac{f}{f^*} = \frac{2K - \frac{K}{N}}{K}$, which gives the desired bound. ■

3.2. *Offline algorithms*

In this section, we propose new polynomial time (offline) approximation algorithms and prove the corresponding performance guarantees. Our results improve upon the bounds of Adler, Gibbons, and Matias (2002) and Dawande, Kumar, and Sriskandarajah (2003). Our algorithms make use of the 2-approximation Linear Programming Rounding (LPR) algorithm proposed earlier by Dawande, Kumar, and Sriskandarajah (2003). Therefore, for completeness we first provide a brief description of this algorithm.

3.2.1. *A 2-approximation algorithm (Dawande, Kumar, and Sriskandarajah, 2003)*

Denote the linear programming (LP) relaxation of $IP_{\min}$ by $LP_{\min}$. The scheduling of ads in slots can be viewed as assignments in a bipartite graph, $G(V_1 \cup V_2, E)$, where nodes V_1 represent the ads and the nodes V_2 represent the slots. An edge $(i, j) \in E$ represents the assignment of ad A_i to slot j. Let $\bar{x}$ be an optimum basic feasible solution to $LP_{\min}$ and let g denote the vector of fractional variables of $\bar{x}$. The *residual graph*, R_G, which is the subgraph of G induced by the edges of g is acyclic (see Lemma 4.1 in Dawande, Kumar, and Sriskandarajah, 2003). Each ad node in R_G has degree at least two since the ad frequency is an integer and the residual graph contains only the edges corresponding to the fractional variables. Let $\bar{f}_j$ denote the fullness of slot j in $\bar{x}$.

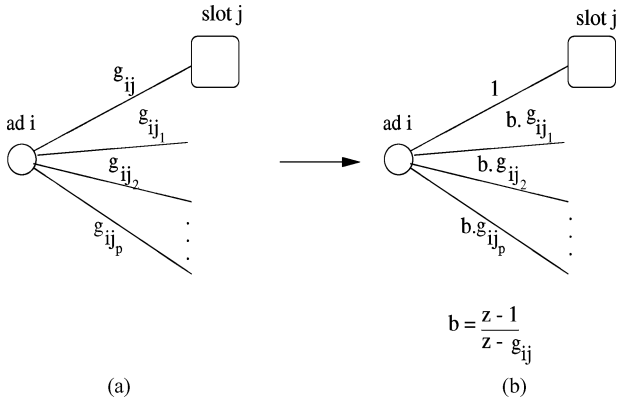


Figure 2. Rounding step for a singleton slot

A slot node of degree 1 in R_G is referred to as a *singleton* slot. A simple rounding step can be used to remove a singleton slot from R_G. Consider a singleton slot j and let edge (i, j), where node i corresponds to ad A_i, be the single edge incident to slot j. Let $g_{ij} = \bar{x}_{ij}$ Let $g_{ij_1}, g_{i,j_2} \ldots, g_{i,j_p}$ be the values corresponding to other edges in R_G incident on node i (see Figure 2(a)).

Note that the quantity $g_{ij} + \sum_{k=1}^{p} g_{ij_k}$ is an integer, say z, since the frequency w_i of ad A_i is an integer. The rounding step sets $g_{ij} = 1$ and $g_{ij_k} = (\frac{z-1}{z-g_{ij}})g_{ij_k}, k = 1, \ldots, p$ (Figure 2(b)). Observe that the frequency of ad A_i remains unchanged in the solution after this rounding step. Moreover, the singleton slot j disappears from R_G since there are no fractional edges incident on it. Thus, the rounding step on a singleton slot node removes at least one edge from the residual graph at the expense of increasing the fullness of slot j. The updated residual graph continues to be acyclic and hence there exists at least one singleton slot. The rounding step thus ensures the existence of a singleton slot as the algorithm proceeds (Dawande, Kumar, and Sriskandarajah, 2003). The description of the LPR algorithm follows.

Algorithm LPR (Dawande, Kumar, and Sriskandarajah, 2003)

Step 1: Obtain a vertex solution $\bar{x}$ to the $LP_{\min}$. Construct the residual graph R_G based on the fractional variables.

Step 2: Find a singleton slot node in R_G and perform the rounding step as shown in Figure 2. Update R_G. Repeat Step 2 until R_G is empty (i.e. R_G has no edges).

Let $s_{\max} = \max_{1 \le i \le n} s_i$. Recall that f^* is the optimum solution value of $IP_{\min}$. In Dawande, Kumar, and Sriskandarajah (2003), it is shown that algorithm LPR provides a feasible solution to the MINSPACE problem, with objective function value f_{LPR} satisfying the following upper bound:

$$f_{LPR} < f^* + s_{\max} \tag{4}$$

Since $s_{\max} < f^*$, it follows that $f_{LPR} < 2f^*$ and hence LPR is a 2-approximation for the MINSPACE problem.

3.2.2. A $(1+\frac{1}{\sqrt{2}})$-approximation algorithm

In this subsection, we describe an improved algorithm, termed MIN1. The idea behind this algorithm is as follows: it first constructs a solution H_1 using the LPR algorithm. Then, the algorithm decomposes the set of ads based on their sizes into two subsets: $\underline{s} = \{A_i | S_i \geq \frac{S_{\max}}{\sqrt{2}}\}$ and $\bar{s} = \{A_i \mid S_i < \frac{S_{\max}}{\sqrt{2}}\}$, where $s_{\max} = \max_{1 \leq i \leq n} s_i$. It tries to schedule *all* the ads from set $\underline{s}$ in such a way that none of the slots have more than one ad. If successful, then the algorithm fixes the corresponding variables in $IP_{\min}$ and solves the remainder of the problem by the LPR algorithm to obtain another solution H_2. An appealing property of this decomposition is that at least one of the two solutions is always within a factor of $1+\frac{1}{\sqrt{2}}$ from the optimal solution. Detailed steps and the analysis of the algorithm follow. We denote the objective function value of the solution of the algorithm by f.

Algorithm MIN1

Step 1: Given $s_i, w_i, i = 1, 2, \ldots, n$ and the number of slots N.

Step 2: Run the LPR algorithm on the optimum solution of $LP_{\min}$. Denote the solution by H_1 and its objective function value by h_1. If $\sum_{i:A_i \in \underline{s}} w_i > N$, then choose H_1 as the solution of the algorithm (i.e. $f = h_1$) and terminate. Otherwise go to Step 3.

Step 3: Starting with the complete set of ads, we construct a new solution. Schedule all the ads in set $\underline{s}$, in such a way that none of the slots has more than one ad. Since $\sum_{i:A_i \in \underline{s}} w_i \leq N$, such a schedule is trivially obtained. Solve $LP_{\min}$ with x_{ij} variables for ads $A_i \in \underline{s}$ already fixed using this schedule. Run algorithm LPR on the optimum solution of $LP_{\min}$ and denote the resulting solution by H_2. Let h_2 be the objective function value of H_2. Choose the better of H_1 and H_2 as the solution of the algorithm. That is, $f = \min\{h_1, h_2\}$.

Both Step 2 and Step 3 require time $O(n^3 N^3 L)$ where L is the length of the binary encoding of $LP_{\min}$ (Dawande, Kumar, and Sriskandarajah, 2003; Martin, 1999).

Theorem 2. The performance bound of MIN1 is $\frac{f}{f^*} \leq 1+\frac{1}{\sqrt{2}}$ and this bound is tight.

Proof. Either $f^* \geq \sqrt{2}s_{\max}$ or $f^* < \sqrt{2}s_{\max}$. We consider the following two cases.

Case 1: $f^* \geq \sqrt{2}s_{\max}$

From (4), the solution of the LPR algorithm satisfies $f_{LPR} < f^* + s_{\max}$. Since $f^* \geq \sqrt{2}s_{\max}$, we have $f \leq h_1 = f_{LPR} < f^*(1+\frac{1}{\sqrt{2}})$.

Case 2: $f^* < \sqrt{2}s_{\max}$

If $\sum_{A_i \in \underline{s}} w_i > N$, then any optimal solution has at least one slot with more than one ad from $\underline{s}$ and consequently $f^* \geq \sqrt{2}s_{\max}$ which is a contradiction. Therefore, $\sum_{A_i \in \underline{s}} w_i \leq N$ and in any optimal solution, each slot has at most one ad from $\underline{s}$. Thus, the algorithm executes Step 3. Without loss of generality, we assume that in the solution H_2 all the ads from $\underline{s}$ are scheduled in a similar way as in the optimal solution and hence the solution f_{lp} to the linear programming relaxation in Step 3 satisfies $f_{lp} \leq f^*$. Algorithm LPR then provides a feasible solution with objective value at most $f^* + \max_{i:A_i \in \bar{s}} s_i$. Since the maximum size of ads in $\bar{s}$ is always less than $\frac{s_{\max}}{\sqrt{2}}$, we have $h_2 < f^* + \frac{s_{\max}}{\sqrt{2}}$. Using $f^* \geq s_{\max}$, we have $f \leq h_2 < f^*(1+\frac{1}{\sqrt{2}})$. ■

Table 2. An example to illustrate the tightness of the bound for algorithm MIN1

Ad A_i	A_1	A_2	A_3
s_i	$1+\epsilon$	1	$\frac{1}{\sqrt{2}}$
w_i	1	1	1

In order to show that the bound is tight, consider the example given in Table 2. We have 3 ads $A_i, i = 1, \ldots, 3$ and $N = 3$. We have $s_{\max} = 1 + \epsilon$, $\underline{s}, = \{A_1, A_2\}$, $\bar{s} = \{A_3\}$. An optimal solution of the LP relaxation at Step 2 has objective function value $\frac{1+2\sqrt{2}+\sqrt{2}\epsilon}{3\sqrt{2}}$. The values of the variables are: $\bar{x}_{11} = \frac{1+2\sqrt{2}+\sqrt{2}\epsilon}{3\sqrt{2}(1+\epsilon)}$, $\bar{x}_{12} = 0$, $\bar{x}_{13} = \frac{\sqrt{2}-1+2\sqrt{2}\epsilon}{3\sqrt{2}(1+\epsilon)}$, $\bar{x}_{21} = 0$, $\bar{x}_{22} = \frac{1+2\sqrt{2}+\sqrt{2}\epsilon}{3\sqrt{2}}$, $\bar{x}_{23} = \frac{\sqrt{2}-1-\sqrt{2}\epsilon}{3\sqrt{2}}$, $\bar{x}_{31} = 0$, $\bar{x}_{32} = 0$, $\bar{x}_{33} = 1$. The residual graph. R_G, then has four edges: (1, 1), (1, 3), (2, 2) and (2, 3). If algorithm LPR chooses to round variable $\bar{x}_{11}$ corresponding to edge (1, 1) to 1 and variable $\bar{x}_{23}$ corresponding to edge (2, 3) to 1, the value of the feasible solution is $h_1 = 1 + \frac{1}{\sqrt{2}}$. An optimal solution of LP relaxation in Step 3 is $\bar{x}_{11} = 1$, $\bar{x}_{12} = 0$, $\bar{x}_{13} = 0$, $\bar{x}_{21} = 0$, $\bar{x}_{22} = 1$, $\bar{x}_{23} = 0$, $\bar{x}_{31} = 0$, $\bar{x}_{32} = \sqrt{2}\epsilon$, $\bar{x}_{33} = 1 - \sqrt{2}\epsilon$. The residual graph, R_G, then has two edges: (3, 2) and (3, 3). If algorithm LPR chooses to round variable $\bar{x}_{32}$ corresponding to edge (3, 2) to 1, the value of the feasible solution is $h_2 = 1 + \frac{1}{\sqrt{2}}$. Therefore, $f = \min_{1 \le i \le 2} h_i = 1 + \frac{1}{\sqrt{2}}$. The optimum solution value is $f^* = 1 + \epsilon$ by placing ad A_1 in slot 1, ad A_2 in slot 2 and ad A_3 in slot 3. Then, $\frac{f}{f^*} = \frac{1+\frac{1}{\sqrt{2}}}{1+\epsilon}$. Thus, $\frac{f}{f^*} \to 1 + \frac{1}{\sqrt{2}}$ as $\epsilon \to 0$. ■

3.2.3. A $\frac{3}{2}$-approximation algorithm

The main idea behind our next algorithm, MIN2, is easy to explain: After sorting the ads in non-increasing order of their size, it considers a subset of ads $\{A_1, \ldots, A_k\} \subseteq A$ with $\bar{w} = \sum_{i=1}^{k} w_i \le N$ and $\bar{w} + w_{k+1} > N$. Then, k solutions $H_i, i = 1, \ldots, k$ are constructed as follows: in H_i, ads $A_q, q = 1, \ldots, i$ are scheduled with one ad per slot to any $w_1 + w_2 + \cdots w_i$ slots. The variables corresponding to this schedule are fixed in the LP relaxation $LP_{\min}$ and the remainder of the problem is solved by running the LPR algorithm on the solution of $LP_{\min}$. The interesting property of this procedure is that at least one of these k solutions is within a factor of $\frac{3}{2}$ from the optimum. The description and the analysis of this algorithm follows.

Algorithm MIN2

Step 1: Given $s_i, w_i, i = 1, 2, \ldots, n$ and the number of slots N, sort the ads in the non-increasing order of their size. Re-index the ads in the sorted order so that $s_i \ge s_2 \cdots \ge s_n$.

Step 2: Let A_k be the ad for which $\sum_{i=1}^{k} w_i \le N$ and $\sum_{i=1}^{k+1} w_i > N$. Construct k different solutions $H_i, i = 1, 2, \ldots, k$ as follows: For solution H_i, schedule the ads $A_q, q = 1, \ldots, i$ with one ad per slot to any $w_1 + w_2 + \cdots w_i$ slots. Since $w_1 + w_2 + \cdots + w_i < N$, such a schedule is trivially obtained. Solve the LP relaxation $LP_{\min}$ with the variables $x_{qj}, q = 1, \ldots, i; j = 1, \ldots, N$ fixed using the schedule above. Let f_{lp}^i denote the objective function value of this LP relaxation. Run the LPR algorithm on the solution corresponding to f_{lp}^i and store its solution as H_i. Let h_i denote the objective function value of H_i.

Step 3: Compare all the stored solutions $H_i, i = 1, 2, \ldots, k$ and select the best solution as the heuristic solution f, i.e., $f = \min_{1 \le i \le k} h_i$.

Table 3. An example to illustrate the tightness of the bound for algorithm MIN2

Ad A_i	A_1	A_2	A_3	A_4
s_i	1	$1-\epsilon$	$\frac{1}{2}$	$\frac{1}{2}$
w_i	1	1	1	1

The running time of the algorithm is dominated by that of Step 2 and is $O(n^4N^3L)$ where L is the length of the binary encoding of $LP_{\min}$ (Dawande, Kumar, and Sriskandarajah, 2003; Martin, 1999).

Theorem 3. The performance bound of algorithm MIN2 is $\frac{f}{f^*} \leq \frac{3}{2}$ *and this bound is tight.*

Proof. Without loss of generality, we can assume that ad A_1 is scheduled in the first w_1 slots. Then, $f^1_{lp} = f_{lp}$. Using (4), $h_1 \leq f^1_{lp} + s_2 = f_{lp} + s_2 \leq f^* + s_2$. If $f^* \geq 2s_2$, we have the required bound for H_1. Otherwise, copies of ads A_1 and A_2 cannot share a common slot in the optimal solution. Thus, $w_1 + w_2 \leq N$ and consequently $f^2_{lp} = f_{lp}$ with $h_2 \leq f^* + s_3$. Again, if $f^* \geq 2s_3$, we have the required result for H_2. Otherwise, we continue the argument using solution H_3. Finally, solution H_k satisfies $h_k \leq f^* + s_{k+1}$. If $f^* < 2s_{k+1}$, no pair of ads A_i, $i = 1, \ldots, k+1$ can share a common slot in the optimal solution. It follows that $\sum_{i=1}^{k+1} w_i \leq N$ which is a contradiction. Hence, $f^* \geq 2s_{k+1}$ and we have the required bound.

To prove the tightness of this bound, consider the example given in Table 3. We have 4 ads A_i, $i = 1, \ldots, 4$ and $N = 3$. An optimal solution to the LP relaxation in Step 2 is $\bar{x}_{11} = 1$, $\bar{x}_{12} = 0$, $\bar{x}_{13} = 0$, $\bar{x}_{21} = 0$, $\bar{x}_{22} = 1$, $\bar{x}_{23} = 0$, $\bar{x}_{31} = 0$, $\bar{x}_{32} = 0$, $\bar{x}_{33} = 1$, $\bar{x}_{41} = 0$, $\bar{x}_{42} = 2\epsilon$, $\bar{x}_{43} = 1 - 2\epsilon$. The residual graph, R_G, has two edges: (4,2) and (4,3). If algorithm LPR chooses to round variable $\bar{x}_{42}$ corresponding to edge (4, 2) to 1, the value of the feasible solution is $h_1 = \frac{3}{2} - \epsilon$. Similarly, $h_2 = \frac{3}{2} - \epsilon$ and $h_3 = \frac{3}{2} - \epsilon$. Therefore, $f = \min_{1 \leq i \leq 3} h_i = \frac{3}{2} - \epsilon$. The optimum solution value is $f^* = 1$ by placing ad A_1 in slot 1, ad A_2 in slot 2, ads A_3 and A_4 in slot 3. Then, $\frac{f}{f^*} = \frac{3}{2} - \epsilon$. ■

Although Algorithm MIN1 has an inferior performance guarantee (compared to that of MIN2), it is a basic algorithm that demonstrates the advantage of decomposing the set of ads into two subsets: a set with large-size ads (easy to schedule) and a set with small-size ads (difficult to schedule). Exploiting this idea further may lead to better approximation algorithms for the MINSPACE problem.

4. CONCLUSIONS

In this paper, we improve upon existing approximation algorithms for the MINSPACE problem (Adler, Gibbons, and Matias, 2002; Dawande, Kumar, and Sriskandarajah, 2003). Our results include (i) an online algorithm with a performance guarantee of $2 - \frac{1}{N}$, where N is the number of slots, and (ii) two (offline) algorithms with performance guarantees of $1 + \frac{1}{\sqrt{2}}$ and $\frac{3}{2}$, respectively. These bounds improve over the current best bound of 2 (Adler, Gibbons, and Matias, 2002; Dawande, Kumar, and Sriskandarajah, 2003).

Since our analysis for all the algorithms is tight, improving the bounds will require the design of new algorithms. We feel that this is a challenging and promising direction for future work. As is generally the case, we believe that the performance of these algorithms on real-world data might be significantly better than their worst case guarantees. Therefore, a possible direction for future research is a study of their average case performance under appropriate assumptions on the type of ads that need to be placed on a web page.

REFERENCES

Adler, M., P. B. Gibbons, and Y. Matias, "Scheduling space-sharing for internet advertising," *Journal of Scheduling*, **5**(2), 103–119 (2002).

Albers, S., "Better bounds for online scheduling," *SIAM Journal on Computing*, **29**(2), 459–473 (1999).

Bartal, Y., A. Fiat, H. Karloff, and R. Vohra, "New algorithms for an ancient scheduling problem," *Journal of Computer and System Sciences*, **51**, 359–366 (1995).

Chandrasekaran, R., B. Chen, G. Galambos, P. R. Narayanan, A. van Vilet, and G. J. Woeginger, "A note on' and on-line scheduling heuristic with better worst case ratio than Graham's list scheduling," *SIAM Journal on Computing*, **26**(3), 870–872 (1997).

Coffman, Jr., E. G., M. R. Garey, and D. S. Johnson, "An application of bin-packing to multiprocessor scheduling," *SIAM Journal of Computing*, **7**(1), 1–17 (1978).

Dawande, M., S. Kumar, and C. Sriskandarajah, "Performance bounds of algorithms for scheduling advertisements on a web page," *Journal of Scheduling*, **6**, 373–393 (2003).

Fleischer, R. and M. Wahl, "On-line scheduling revisited," *Journal of Scheduling*, **3**(6), 343–353 (2000).

Friesen, D. K., "Tighter bounds for multifit processor scheduling algorithm," *SIAM Journal of Computing*, **13**, 170–181 (1984).

Friesen, D. K. and M. A. Langston, "Evaluation of a multifit-based scheduling algorithm," *Journal of Algorithms*, **7**, 35–59 (1986).

Fruend, A. and J. Naor, "Approximating the advertisement placement problem," *Journal of Scheduling*, **7**, 365–374 (2004).

Galambos, G. and G. J. Woeginger, "An on-line scheduling heuristic with better worst case ratio than Graham's list scheduling," *SIAM Journal on Computing*, **22**(2), 349–355 (1993).

Graham, R. L., "Bounds on multiprocessing timing anomalies," *SIAM Journal of Applied Mathematics*, **17**, 416–429 (1969).

Hochbaum, D. S. and D. B. Shmoys, "Using dual approximation algorithms for scheduling problems: Theoretical and practical results," *Journal of the Association for Computing Machinery*, **34**, 144–162 (1987).

Hyland, T. "Why Internet advertising?" in *Webvertising: The Ultimate Internet Advertising Guide*, SCN Education B. V., Friedrich Viewag & Sohn, 2000, pp. 13–18.

Interactive Advertising Bureau, http://www.iab.net.

Karger, D. R., S. J. Phillips, and E. Torng, "A better algorithm for an ancient scheduling problem," *Journal of Algorithms*, **20**(2), 400–430 (1996).

Martin, R. K., *Large Scale Linear and Integer Optimization*, Kluwer, Massachusetts, 1999.

Puetz, J., "A revolution in our midst," *Satellite Communications*, Atlanta, **24**, 38 (2000).