Wireless Networks 10, 157–168, 2004

Broadcast Disks with Polynomial Cost Functions

AMOTZ BAR-NOY *
Computer and Information Science Department, Brooklyn College, 2900 Bedford Avenue Brooklyn, NY 11210, USA

BOAZ PATT-SHAMIR and IGOR ZIPER
Department of Electrical Engineering, Tel Aviv University, Tel Aviv 69978, Israel

Abstract. In broadcast disks systems, information is broadcasted in a shared medium. When a client needs an item from the disk, it waits until that item is broadcasted. Broadcast disks systems are particularly attractive in settings where the potential customers have a highly-asymmetric communication capabilities, i.e., receiving is significantly cheaper than transmitting. This is the case with satellite networks, mobile hosts in wireless networks, and Teletext system.

The fundamental algorithmic problem for such systems is to determine the broadcast schedule based on the demand probability of items, and the cost incurred to the system by clients waiting. The goal is to minimize the mean access cost of a random client. Typically, it was assumed that the access cost is proportional to the waiting time. In this paper, we ask what are the best broadcast schedules for access costs which are arbitrary polynomials in the waiting time. These may serve as reasonable representations of reality in many cases, where the "patience" of a client is not necessarily proportional to its waiting time.

We present an asymptotically optimal algorithm for a fractional model, where the bandwidth may be divided to allow for fractional concurrent broadcasting. This algorithm, besides being justified in its own right, also serves as a lower bound against which we test known discrete algorithms. We show that the Greedy algorithm has the best performance in most cases. Then we show that the performance of other algorithms deteriorate exponentially with the degree of the cost polynomial and approaches the fractional solution for sub-linear cost. Finally, we study the quality of approximating the greedy schedule by a finite schedule.

Keywords: broadcast disks, scheduling, satellite networks

1. Introduction

The idea of *broadcast disks*, introduced by Franklin et al. [1,2], has gained considerable popularity lately (see, e.g., [8,9,18,20,24–26]). Intuitively, the goal is to implement push-systems using a shared communication medium. Specifically, it is assumed that a set of customers have a receive-only access to a shared medium, whose only transmitting source is a server. The server maintains a set of information items called pages. The pages are broadcasted in some order called schedule. The idea is that when a customer wishes to access a page, it starts listening until the desired page is broadcasted. Broadcast disks systems are particularly attractive in settings where the potential customers have a highly-asymmetric communication capabilities, i.e., receiving is significantly cheaper than transmitting. This is the case with satellite networks, mobile hosts in wireless networks, and Teletext system [3,4].

The problem of customers waiting for a long time is inherent to the model of broadcast disks (by the pigeon-hole principle, if there are m pages, then there must be at least one page with at least m time units between two successive instances of its broadcast). In absolute terms, this problem can solved by using faster shared media, but in relative terms, better solutions are required. Indeed, it was shown that some a-priory knowledge of the access patterns by the customers could lead to a better performance. In order to evaluate the performance of scheduling schemes, it is assumed that the access patterns of customers can be modeled by a probability distribution over the pages; and that the "damage" incurred to the system by waiting customers can be modeled by a cost function, assigning a real number to the waiting time intervals. Assuming that customers statistics can be gathered, and that the cost function can be estimated, the natural objective in this model is to find the schedule that minimizes the *expected cost* of a random customer. This question is the main motivation of this paper.

Specifically, we are interested in the following variant of the problem. Suppose that the cost of a customer waiting i consecutive time units (slots) is some function $c(i)$. Most previous work concentrated on the case were $c(i) = i$, i.e., the cost is proportional to the waiting time. In this paper we are motivated by the assumption that in reality, there are several types of customers. In some settings, the patience of the customers may run out quickly, while in others, customers may have a lot of patience. For example, the third slot might be more expensive than the first one, since the rate in which customers are switching service may grow with time. In other cases, however, it may reasonable to assume that the cost of the first time unit is much more than the cost of the tenth time unit. We model this fact with cost functions which are super-linear or sub-linear, respectively. In particular, we study cases where $c(i)$ is (i) polynomial in i with degree at least 1 (super-linearity), and (ii) polynomial in i with degree less than 1 (sub-linearity).

* Corresponding author.
E-mail: amotz@sci.brooklyn.cuny.edu

Our contributions

First, we analyze the known fractional relaxation to a general polynomial cost functions. The idea behind the fractional relaxation is to consider the problem where the bandwidth constraint is imposed only on the whole cycle, thus allowing at each instant more than one page to be served. This relaxation allows us to study the optimal bandwidth allocation and spacing between consecutive instances of a given page, without paying attention to the particular way other pages are scheduled. Obviously, a discrete schedule (which is our main interest) is also a solution in the fractional model. The converse, however, does not hold in general. We show how to solve the fractional relaxations asymptotically optimally for any polynomial cost function. The fractional solutions provide us with convenient lower bounds for the discrete scheduling problem we consider.

Next, we evaluate the quality of some of the scheduling algorithms proposed in the literature. We compare the performance of the following algorithms for a polynomial cost function:

Random: choose the next page to broadcast independently at random, with probabilities proportional to the optimal bandwidths in the fractional model.

Halving: round the fractional solution to powers of $1/2$, which are easy to arrange in a discrete schedule.

Fibonacci (a.k.a *Golden Ratio*): approximate the fractional solution using a golden-ratio sequence [8,22].

Greedy: broadcast, at each time step, the page which will incur the highest cost if not broadcasted.

We show that the analytical performance bounds (the ratio of the discrete approximation to the fractional solution) deteriorates exponentially with the degree of the cost polynomial. This fact is supported by extensive simulation results for the algorithms. However, we show that there is a substantial difference between the performance of the algorithms (since the exponent bases differ significantly): the Greedy algorithm outperforms all others in almost all cases. The variables we examine are the size of the database (i.e., how many pages does the server have to transmit), and the degree of the cost polynomial (representing a measure of how patient or nervous is a typical customer).

Guided by these results, we study more closely the Greedy algorithm. The main problem with the Greedy algorithm is that it may be expensive to implement: at each step, one has to scan the whole database in order to find the right item to transmit. A simple approximation to this computation-intensive approach is to compute the schedule up to some point, and then repeat it using table lookups. The natural question in this case is what should be the cycle length. The results turn out to be somewhat surprising: the cost incurred by a truncated greedy schedule is *not* a monotonic function of the cycle length. Roughly speaking, for a cost function $c(i) = i^\alpha$ and a database with m pages, it seems that the best strategy is to truncate the schedule at the cheapest point in the interval $[10\alpha m, 10\alpha m + m]$.

Paper organization

The rest of this paper is organized as follows. In section 2 we give a brief description of related work. Section 3 describes the model and introduces some terminology. Section 4 explains how to find the lower bound for the overall mean cost for general polynomial cost functions via the fractional model. Section 5 presents the evaluation of the various algorithms for polynomial cost functions.

2. Related work

There are many variants of the broadcast disks problem and there are many other problems that share similar objectives. In what follows we outline some of them. Ammar and Wong in [3,4] study extensively the problem of minimizing the mean response time in Teletext systems. This problem is essentially the same as minimizing the mean waiting time in broadcast disks. They show that an optimal cyclic schedule exists and their results implicitly contain a randomized 2-approximation algorithm for this problem. This algorithm broadcasts a page i with probability proportional to $\sqrt{p_i}$, where p_i is the probability that a customer accesses page i. In [8], Bar-Noy et al. discuss a more general problem called the maintenance problem. one of their contribution was using the properties of the golden ratio sequence for building efficient broadcast schedules. The method of designing broadcast schedules is considered by Su and Tassiulas [24] in the context of the broadcast disks. They empirically test the performance of a class of greedy heuristics for this problem. They report that the greedy policy which selects the page with largest *mean aggregate delay* for broadcasting has the best performance in most cases. Furthermore, this policy produces schedules with mean response time which is very close to the lower bound on obtained by a fractional solution [3,4]. The greedy rule that is used by our algorithm is similar to the one in the heuristic of [6] and is the same as the *mean aggregate delay* rule described in [24].

Anily et al. [6] consider the problem in the setting of the maintenance problem, where the cost is linear in the waiting time. In [7], the same authors give an optimal solutions for most cases for a 3-machine case in the same model (which is equivalent to a database of size 3 in our setting), and give approximation algorithms for the remaining cases. In a recent paper, Anily and Bramel consider arbitrary convex cost functions [5]. They prove that there always exists an optimal cyclic schedule, and give a thorough treatment for the case of two machines.

Additional papers containing analysis of similar types of problems are [11–15,17,23,27].

3. Model and preliminaries

In this section we formally describe the model under consideration, and define the terminology to be used in the rest of the paper.

3.1. Basic model (discrete)

The *server database* contains m information items called *pages*, denoted by $A_1, A_2, \ldots, A_m$. We assume that all pages are of the same size. Furthermore, we assume that for each page A_i there is a *demand probability* p_i, such that $\sum_{i=1}^{m} p_i = 1$. The number p_i represents the probability that a random customer wants to get A_i.

We assume a slotted time model in which the server broadcasts one page per time slot. The order in which the server broadcasts its pages is called the *server's schedule.* We will be interested in *cyclic schedules*, defined by a repeated finite segment. We denote such a segment by $S = S_0, S_1, S_2, \ldots, S_{N-1}$, where $S_t \in \{A_1, A_2, \ldots, A_m\}$ for all t. Henceforth, we consider only cyclic schedules (this is justified by the result of Anily and Bramel [5] that there always exists an optimal cyclic schedule). The letter N is reserved to denote the length of the cycle in the schedule under consideration. An appearance of a page in the schedule is referred to as an *instance* of the item. The *spacing* of a page A_i is the time between two consecutive instances of A_i. We say that $s_i(t) = x$ if A_i is broadcasted at time $t + x$ but not in times $t + 1, \ldots, t + x - 1$. In other words, $s_i(t)$ is the number of time slots the customer starting to wait at time t has to wait until page A_i is broadcasted. The *average frequency* of a given page in a schedule with cycle length N is the number of its instances in any interval of N time units, divided by N.

3.2. Cost models

The natural way to model the cost of a schedule is to assume that we are given a *cost function* $c(t)$ which represents the cost of waiting t time slots. To calculate the expected cost of a given schedule S, we assume that customers arrive at the beginning of a random time slot, i.e., for a schedule with cycle length N, all N slots are equally likely (a similar approach is taken in [8]). We first compute C_i, the expected cost incurred by a random customer, given that the customer is waiting for page A_i with a given cost function c:

$$C_i(S) = \frac{1}{N} \sum_{t=0}^{N-1} c\big(s_i(t)\big). \tag{1}$$

Hence the expected cost for a random customer is given by the following formula:

$$C(S) = \sum_{i=1}^{m} p_i C_i(S) = \sum_{i=1}^{m} \left(\frac{p_i}{N} \sum_{t=0}^{N-1} c\big(s_i(t)\big) \right). \tag{2}$$

A special case of interest is when all pages are exactly evenly spaced. In this case we denote the spacing for page A_i by s_i, and we get the following definition for the expected cost:

$$C_i(S) = \frac{1}{s_i} \sum_{t=1}^{s_i} c(t), \tag{3}$$

$$C(S) = \sum_{i=1}^{m} \left(\frac{p_i}{s_i} \sum_{t=1}^{s_i} c(t) \right). \tag{4}$$

A schedule S is called *optimal* if it minimizes $C(S)$.

In many cases, it is more convenient to work with a *gap function*, which is the sum of the cost function $c(t)$ over an interval of time. Formally, we define the gap function $g(s)$ by the following equation:

$$g(s) = \sum_{t=1}^{s} c(t). \tag{5}$$

Note that a gap function uniquely defines a cost function, since $c(t) = g(t) - g(t-1)$. That is, defining a cost function is equivalent to defining a gap function. Using the gap function, it is simpler to express the cost of a given schedule. We give only the the definition for equal-spacing schedules, where the space between successive instances of A_i is s_i.

$$C(S) = \sum_{i=1}^{m} p_i \frac{g(s_i)}{s_i}. \tag{6}$$

4. Analysis of the fractional model

In this section we consider a different variant of the model presented in section 3, called the *fractional model*. The fractional model is a relaxation of the discrete model, since it assumes that a server may broadcast a fraction of pages. This model serves as a starting point of some of the algorithms we study. It also uses as a convenient lower bound for the cost of schedules in our main (discrete) model.

The fractional model is defined as follows. We are given a database with demand probabilities as before. Here, however, any number of pages can be broadcasted in a time-slot, as long as the total number of broadcasts is bounded by the cycle size. The variables in the fractional model are real numbers N and $n_1, \ldots, n_m$, where N is the length of the cycle, and n_i denotes the number of times page A_i is broadcasted during the cycle, for $i = 1, \ldots, m$. There is a single constraint that $\sum_{i=1}^{m} n_i \leqslant N$. The objective is to minimize the expected cost of the schedule. The basic property of an optimal fractional schedule is stated in the following lemma.

Lemma 1. Let $c(t)$ be any increasing cost function. Given a fraction for page A_i, the optimal broadcast schedule with minimum expected cost results when the instances of each page are equally spaced.

Proof. Let S be an optimal schedule. Suppose, for contradiction, that page A_i is non-equally spaced. We prove the lemma by constructing a schedule S' with $C(S') < C(S)$.

Consider the instances of page A_i. Since A_i is not equally spaced, we can find three instances such that the second instance appears s_1 time after the first, and the third instance appears s_2 time after the second, where $s_1 > s_2$ (see figure 1). Let $s = (s_1 + s_2)/2$. Without loss of generality we can write that $s_1 = s + a$ and $s_2 = s - a$, where $a > 0$. Let $C_i(x)$ denote the cost incurred by a random customer, given that the customer wants page A_i and that he arrived in a time

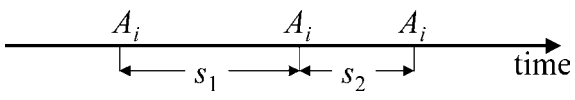

Figure 1. The schedule S considered in the proof of lemma 1. Page A_i appears with two consecutive different spacings.

interval of length x between two occurrences of A_i. Then $C_i(x) = \int_1^x c(t)\,\mathrm{d}t$.

Let C_{rest} be the cost incurred by a random customer waiting for all pages except for the two instances of A_i in the interval we consider. With this notation, the expected cost of S is

$$\begin{aligned} C(S) &= C_i(s_1) + C_i(s_2) + C_{\text{rest}} \\ &= \int_1^s c(t)\,\mathrm{d}t + \int_{s+1}^{s+a} c(t)\,\mathrm{d}t + \int_1^s c(t)\,\mathrm{d}t \\ &\quad - \int_{s-a+1}^{s} c(t)\,\mathrm{d}t + C_{\text{rest}} \\ &= 2\int_1^s c(t)\,\mathrm{d}t + \int_{s+1}^{s+a} c(t)\,\mathrm{d}t - \int_{s-a+1}^{s} c(t)\,\mathrm{d}t + C_{\text{rest}} \\ &> 2\int_1^s c(t)\,\mathrm{d}t + C_{\text{rest}}. \end{aligned}$$

The last inequality is true since the length of interval $[s+1 : s+a]$ equals the length of interval $[s+1-a : s]$ and $c(t)$ is increasing by assumption, and hence $\int_{t=s+1}^{s+a} c(t)\,\mathrm{d}t > \int_{t=s-a+1}^{s} c(t)\,\mathrm{d}t$.

Consider now the schedule S' defined as follows. S' is identical to S except that we move the second instance of page A_i such that that the spacing is s between the first and the second instance, and hence the spacing is also s between the second and third instance. Note that this is possible in the fractional model. The expected cost of schedule S' is as follows:

$$C(S') = C(s) + C(s) + C_{\text{rest}} = 2\int_{t=1}^{s} c(t)\,\mathrm{d}t + C_{\text{rest}}.$$

Clearly, $C(S) > C(S')$, contradicting the optimality of S. □

Note that one can try to convert a fractional schedule to a discrete schedule by assigning spacing s_i for page A_i. If this works, then optimality is guaranteed by lemma 1. The problem is that s_i is not necessarily integral. (This is the reason why a solution in the fractional model is often called *ffractional schedule*.) Furthermore, even if s_i were integral for all i, it may be the case that no discrete schedule with spacing s_i exists. (Consider, for example, $s_1 = 2$ and $s_2 = 3$; there is no way to schedule A_1 exactly every two time units and A_2 exactly every three time units.) Thus, the fractional model serves as a relaxation of the discrete model which enables us to isolate the problem of finding the optimal spacings. Clearly, any discrete schedule incurs at least as much cost as the optimal fractional schedule. In other words, the cost derived from a fractional solution to the fractional model is a lower bound on the cost in the discrete model. We use this observation by calculating ideal spacing values in the fractional model (which may fractional in general), to establish lower bounds on the cost of an optimal schedule in the discrete model.

In the remainder of this section, we find asymptotically optimal schedules for any polynomial cost function with degree greater than 0 (so that the monotonicity of the cost function is maintained).

4.1. Optimal fractional schedule for general linear cost function

Suppose that the cost function is the general polynomial of degree 1, denoted $c(t) = \beta t + \gamma$. By lemma 1, we consider schedules where all instances of A_i are equally spaced with s_i slots between successive instances. We first calculate the expected cost of a random client waiting for page A_i:

$$C_i = \frac{1}{s_i}\sum_{t=1}^{s_i}(\beta t + \gamma) = \beta\frac{s_i+1}{2} + \gamma.$$

Next, we calculate the overall expected cost of schedule S:

$$C(S) = \sum_{i=1}^{m} p_i C_i = \sum_{i=1}^{m} p_i\left(\beta\frac{s_i+1}{2} + \gamma\right). \tag{7}$$

The theorem below provides the basis for an optimal schedule for general linear cost functions.

Theorem 2. In the fractional model, the minimum expected cost is achieved when the frequency q_i of each page A_i satisfies

$$q_i = \frac{\sqrt{p_i}}{\sum_{j=1}^{m}\sqrt{p_j}}.$$

Proof. In the fractional model each page A_i appears in the schedule with frequency $q_i = 1/s_i$. The mean expected cost of schedule S to be paid by a random client is as follows:

$$\begin{aligned} C(S) &= \sum_{i=1}^{m} p_i C_i \\ &= \sum_{i=1}^{m} p_i\left(\beta\frac{s_i+1}{2} + \gamma\right) \\ &= \frac{\beta}{2}\sum_{i=1}^{m} p_i s_i + \frac{\beta}{2}\sum_{i=1}^{m} p_i + \gamma\sum_{i=1}^{m} p_i \\ &= \frac{\beta}{2}\sum_{i=1}^{m}\frac{p_i}{q_i} + \frac{\beta}{2} + \gamma. \end{aligned}$$

Hence

$$\frac{\partial C(S)}{\partial q_1} = \frac{\beta}{2}\cdot\frac{\partial}{\partial q_1}\sum_{i=1}^{m}\frac{p_i}{q_i}. \tag{8}$$

As $\sum_{i=1}^{m} q_i = 1$, only $m-1$ terms of the sum can change simultaneously. Therefore we get:

$$\frac{\partial C(S)}{\partial q_1} = \frac{\beta}{2}\left[\frac{\partial}{\partial q_1}\left(\frac{p_1}{q_1}\right) + \frac{\partial}{\partial q_1}\left(\sum_{i=2}^{m-1}\frac{p_i}{q_i}\right) + \frac{\partial}{\partial q_1}\left(\frac{p_m}{1-\sum_{i=1}^{m-1} q_i}\right)\right]$$
$$= \frac{\beta}{2}\left(-\frac{p_1}{q_1^2} + p_m\left(\frac{1}{1-\sum_{i=1}^{m-1} q_i}\right)^2\right).$$

Since for the optimal q_i values we have $\partial C(S)/\partial q_i = 0$ for all i, we get by rearrangement:

$$\frac{p_1}{q_1^2} = p_m\left(\frac{1}{1-\sum_{i=1}^{m-1} q_i}\right)^2.$$

Similarly for any j we get:

$$\frac{p_j}{q_j^2} = p_m\left(\frac{1}{1-\sum_{i=1}^{m-1}(q_i)}\right)^2.$$

Hence we get for all i, j, that in the optimal solution

$$\frac{q_i}{q_j} = \sqrt{\frac{p_i}{p_j}}.$$

Therefore, the optimal q_i is proportional to $\sqrt{p_i}$. Since $\sum_{i=1}^{m} q_i = 1$, we get for all i:

$$q_i = \frac{\sqrt{p_i}}{\sum_{j=1}^{m}\sqrt{p_j}}.$$

Hence s_i is as follows:

$$s_i = \frac{\sum_{j=1}^{m}\sqrt{p_j}}{\sqrt{p_i}}.$$

□

Thus, for common linear case when the cost function is defined as $c(t) = \beta t + \gamma$ the optimal schedule is obtained when every page A_i is equally spaced with $s_i \approx 1/\sqrt{p_i}$ or when the frequency appearance of page A_i in the schedule is proportional to $\sqrt{p_i}$.

This result complements known results [4,25,28] about the "square root rule" for minimizing waiting times, i.e., for the identity cost function $c(t) = t$.

4.2. Asymptotically optimal solutions for general polynomial cost functions

To deal with general polynomial cost functions, it is more convenient to consider gap functions. We first prove basic connections between cost functions and gap functions.

The following lemma shows how to express cost functions for any polynomial gap functions.

Lemma 3. A gap function $g(s) = s^{\alpha+1}$ is equivalent to a cost function

$$c(t) = \sum_{j=0}^{\alpha}(-1)^{\alpha-j}\binom{\alpha+1}{j}t^j.$$

Proof. By the Binomial Law, we have that

$$g(t-1) = \sum_{j=0}^{\alpha+1}(-1)^{\alpha+1-j}\binom{\alpha+1}{j}t^j.$$

Since a gap function uniquely defines the cost function, we get:

$$\begin{aligned} c(t) &= g(t) - g(t-1) \\ &= t^{\alpha+1} - \sum_{j=0}^{\alpha+1}(-1)^{\alpha+1-j}\binom{\alpha+1}{j}t^j \\ &= -\sum_{j=0}^{\alpha}(-1)^{\alpha+1-j}\binom{\alpha+1}{j}t^j \\ &= \sum_{j=0}^{\alpha}(-1)^{\alpha-j}\binom{\alpha+1}{j}t^j. \end{aligned}$$

□

Using lemma 3, we can find the cost function for any given polynomial gap function. Following are two examples for finding coefficients k_j of the cost function which can replace same gap function.

Together with equation (5), we can find the gap function given the cost function and vice-versa. The theorem below provides the basis for exact optimal schedules for arbitrary polynomial gap or cost functions.

Theorem 4. For $\alpha > 0$, if the gap function is $g(s) = s^{\alpha+1}$, then the optimal schedule is achieved when the spacing between any two successive instances of A_i is

$$s_i = \frac{\sum_{j=1}^{m} p_j^{1/(\alpha+1)}}{p_i^{1/(\alpha+1)}}.$$

The proof of theorem 4 is similar to the proof of theorem 2 and we therefore omit it.

Using theorem 4, we can find asymptotically optimal schedules for *any* polynomial cost functions.

Theorem 5. For $\alpha > 0$, if the gap function is $g(s) = s^{\alpha+1} + O(s^{\alpha})$, then the optimal schedule is achieved when the spacing between any two successive instances of A_i satisfies

$$s_i = \frac{\sum_{j=1}^{m} p_j^{1/(\alpha+1)}}{p_i^{1/(\alpha+1)}} + O\left(p_i^{-1/\alpha}\right).$$

Proof. Let us find the optimal s_i when the gap function is $s^{\alpha+1} + O(s^\alpha)$. By definition, we have:

$$\begin{aligned} C(S) &= \sum_{i=1}^{m} p_i\left(s_i^\alpha + O\left(s_i^{\alpha-1}\right)\right) \\ &= \sum_{i=1}^{m} p_i s_i^\alpha + \sum_{i=1}^{m} O\left(p_i s_i^{\alpha-1}\right) \\ &= \sum_{i=1}^{m} \frac{p_i}{q_i^\alpha} + \sum_{i=1}^{m} O\left(\frac{p_i}{q_i^{\alpha-1}}\right). \end{aligned}$$

For optimal values of q_i, we must have $\partial C(S)/\partial q_i = 0$, for all i. Because $C(S)$ is a polynomial, its derivative is the sum derivatives of its terms.

Using theorem 4, we find the optimal q_i:

$$q_i = \frac{p_i^{1/(\alpha+1)}}{\sum_{j=1}^{m} p_j^{1/(\alpha+1)}} + O\left(p_i^{1/\alpha}\right).$$

And therefore,

$$s_i = \frac{\sum_{j=1}^{m} p_j^{1/(\alpha+1)}}{p_i^{1/(\alpha+1)}} + O\left(p_i^{-1/\alpha}\right).$$

□

Thus, we have a good asymptotic estimate for optimal schedules in the fractional model. This estimate may be used by algorithms for the discrete model, and we shall also use it as a benchmark to measure the performance of discrete schedules.

5. Algorithms in the discrete model

In this section we present results comparing the performance of four algorithms in the discrete model. Our methodology is the following. We assume that the demand probability of pages follow the Zipf distribution. This distribution is a good approximation for "real life" situations (see, e.g., [10,25]). The Zipf distribution, with *access skew coefficient* θ, says that the i-th popular page in the database is accessed with probability proportional to $i^{-\theta}$. Formally, we have the following formula for $1 \leqslant i \leqslant m$:

$$p_i = \frac{(1/i)^\theta}{\sum_{i=1}^{m} (1/i)^\theta}.$$

In the remainder of this section, we give, for each algorithm, a brief description, analytical worst-case upper bounds, and experimental results.

In our simulations, we measure the behavior of the algorithms as a function of the size of the database, and of the cost function. We remark that have also measured the influence of the value of the access skew coefficient θ in the range [0.6, 0.95], and observed that the behavior of our algorithms was independent of θ. The results reported in this section are for fixed $\theta = 0.8$, as recommended by Breslau et al. [10].

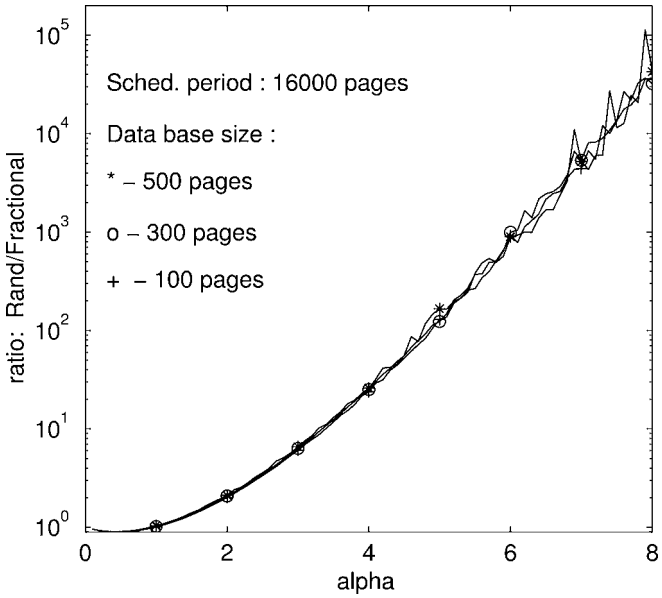

Figure 2. Performance of Random algorithm as function of data base size.

5.1. The random algorithm

The Random algorithm tries to approximate the fractional schedule by using its fractional spacings. Specifically, let $\{q_i\}$ be the frequencies of pages in the optimal fractional schedule (these are simply the bandwidth shares of the pages). The Random algorithm transmits, at each time step, page A_i with probability q_i, so that the *expected* frequency of each page is the *exact* frequency in the fractional solution. It is known that for the linear cost function $c(t) = t$, the expected cost of the schedule generated by the Random algorithm is at most twice the cost of the underlying fractional schedule [8]. It turns out that the performance of Random deteriorates rapidly for cost functions with higher degree polynomial. From the analytic viewpoint, using the same proof technique of [8], we can prove that if $c(x) = x^\alpha$, then the expected cost of the Random algorithm is never more than $(\alpha + 1)!$ (i.e., roughly $((\alpha + 1)/e)^{\alpha+1}$) times worse than the fractional cost. For large values of α, this is a terrible upper bound. However, it is not accidental, as our experiments show. Specifically, let R_{500} be a typical schedule generated by the Random algorithm in a 500-page database. Based on our results, we are able to determine that

$$C(R_{500}) \approx 0.02 \cdot 6.12^\alpha. \tag{9}$$

In other words, even for randomly generated databases, the cost for schedules generated by the Random algorithm is exponentially more expensive than the fractional schedules. This result is independent of the database size (figure 2) and is already evident in relatively short schedules (figure 3).

5.2. The Halving algorithm

The idea in the Halving algorithm is to round the desired page frequencies to the nearest power of $\frac{1}{2}$ from below, since powers of $\frac{1}{2}$ are easily arranged in a discrete schedule. Formally,

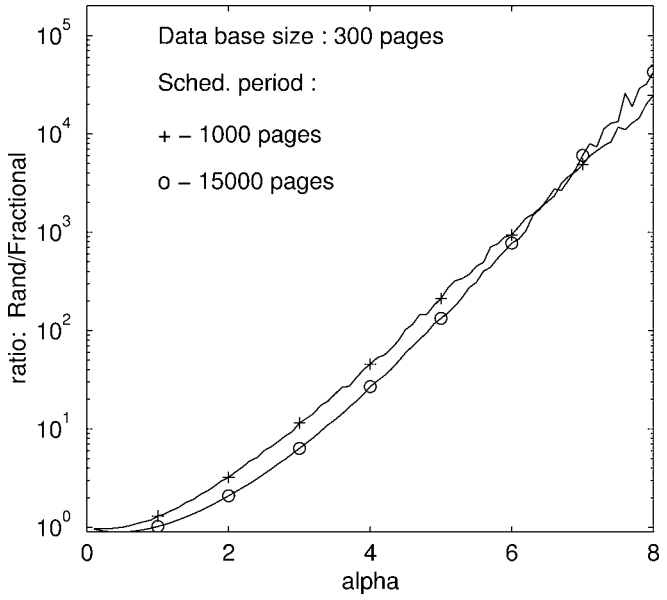

Figure 3. Performance of Random algorithm as function of schedule size.

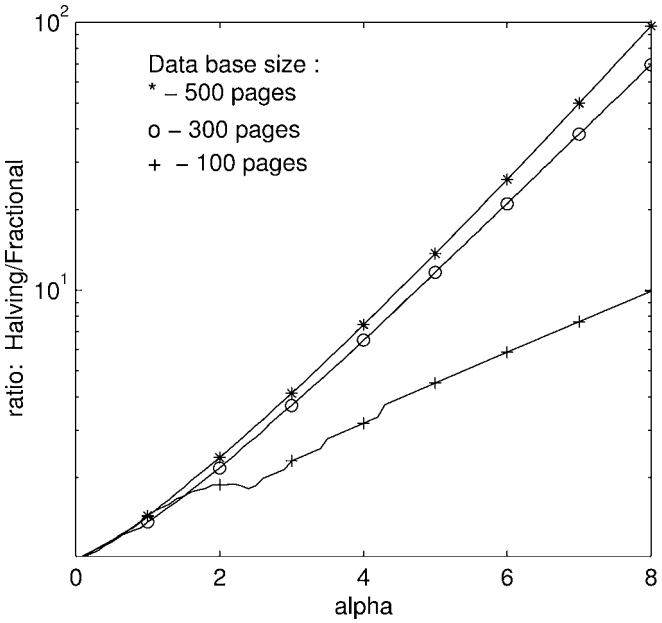

Figure 4. Performance of the Halving algorithm as a function of the database size.

given the optimal page frequencies $\{q_i\}$, page A_i gets a period of $2^{\lceil \log_2(1/q_i) \rceil}$. (To actually generate a schedule, it is sufficient to allocate time slots to pages by decreasing frequencies.) Note that in some cases, one obtains an optimal schedule: this happens when the desired frequencies are already powers of $\frac{1}{2}$. The important point about the Halving algorithm is that even in the worst case, the actual frequency obtained is always more than half of the desired frequency. It follows that in the linear cost model, the Halving algorithm generates schedules whose guaranteed performance is always less than twice the cost of the optimal fractional schedule. However, for general polynomial cost functions, the situation is different.

Let us first see what happens to the worst-case guarantee. Suppose that the gap function is $g(x) = x^{\alpha+1}$. Let s_i^F be the spacing for page A_i in the fractional model. The cost with fractional spacing obtained from the optimal fractional schedule S_F is

$$C(S_F) = \sum_{i=1}^{m} p_i \left(s_i^F\right)^{\alpha}.$$

Let s_i^H be the actual spacing of generated by the Halving algorithm. Since $s_i^H < 2s_i^F$, we get that the cost for schedule S_H obtained from the Halving algorithm is bounded by

$$C(S_H) = \sum_{i=1}^{m} p_i \left(s_i^H\right)^{\alpha} < C(F) \cdot 2^{\alpha}.$$

Note that for small values of α (for example, $\alpha < 1$), this is a reasonable bound. However, for large values of α, this bound is unacceptable. Our experiments indicate that for large values of α, the Halving algorithm indeed exhibits this bad behavior, but not in the same manner the Random algorithm does (see figure 4). Specifically, let H_{500} be a typical schedule generated by the Halving algorithm in a 500-page database. Our results indicate the following parameters for the exponential behavior of H_{500}:

$$C(H_{500}) \approx 0.76 \cdot 1.80^{\alpha}. \tag{10}$$

Moreover, while the performance for a randomly generated database is exponential in the power of the cost function, we find that this phenomenon is far less accentuated in small databases.

5.3. The Fibonacci (golden ratio) algorithm

The Fibonacci algorithm generates a schedule with the same average frequencies as those of the optimal fractional solution. However, unlike the fractional model, the spacing between two consecutive appearances of the same page in the schedule may have three values that are "close" to the optimal periods $(1/q_i)$. The Fibonacci algorithm, also called the *golden ratio sequence*, was developed in [16,19] and is based on an open address hashing method introduced in [21]. For a linear cost function, the Fibonacci algorithm is guaranteed to generate a schedule whose cost is at most $\frac{9}{8}$ times the cost of the fractional-model schedule [8].

In this section we provide an exponential upper bound on the cost of the golden-ratio sequence, by extending (in a non-trivial way) the proof of [8] to the case of a polynomial cost function. However, we are not able to derive an exact expression, and we evaluated the actual exponent numerically.

We start with a brief description of the algorithm. The interested reader can find more details in [8].

Recall that Fibonacci numbers are defined by the following recursive definition: $F_0 = 0$, $F_1 = 1$, and for $k > 1$, $F_k = F_{k-1} + F_{k-2}$. The sequence is 0, 1, 1, 2, 3, 5, 8, 13, ...

We use the following properties of Fibonacci numbers.

Lemma 6.

- For all $k, k' > 0$, $F_k F_{k'} < F_{k+k'}$.
- For all $k > 0$, $F_{k+1}/F_k \leqslant 2$.
- For all $k > 0$, $F_{k+3}/F_k \leqslant 5$.

Consider a given page A_i in a finite schedule. If A_i appears N_i times in the sequence then there are N_i gaps in the schedule (the last gap is computed assuming that the sequence is cyclic).

The following theorem defines the *golden ratio sequence*.

Theorem 7 [16]. Let $g_1, \ldots, g_m$ be frequencies such that $\sum_{i=1}^m g_i = 1$, and let F_k be a Fibonacci number such that $g_i = N_i/F_k = (F_{k-j_i} + S_i)/F_k$ for some $1 \leqslant j_i \leqslant k-2$ and $0 \leqslant S_i < F_{k-j_i-1}$ ($\sum_{i=1}^m N_i = F_k$). Then there exists a sequence of length F_k of the pages $A_1, \ldots, A_m$ such that A_i appears N_i times in the sequence. Furthermore, for each $1 \leqslant i \leqslant m$, there are at most three values for the gaps between two consecutive appearances of A_i in the sequence:

- S_i gaps of value F_{j_i},
- $F_{k-j_i-2} + S_i$ gaps of value F_{j_i+1},
- $F_{k-j_i-1} - S_i$ gaps of value F_{j_i+2}.

Note that there are $S_i + (F_{k-j_i-2} + S_i) + (F_{k-j_i-1} - S_i) = F_{k-j_i} + S_i = N_i$ gaps. Note also that the sum of all gaps is the length of the sequence: $F_{j_i} S_i + F_{j_i+1}(F_{k-j_i-2} + S_i) + F_{j_i+2}(F_{k-j_i-1} - S_i) = F_k$ (see [8]).

The golden ratio sequence: Let $q_1, \ldots, q_m$ denote frequencies ($q_i = 1/s_i$, where $s_1, \ldots, s_m$ is the optimal solution received by theorem 4) such that $\sum_{i=1}^m q_i = 1$. Let F_k be a Fibonacci number. Define $N_i = \lfloor q_i F_k \rfloor$ or $N_i = \lceil q_i F_k \rceil$ such that $\sum_{i=1}^m N_i = F_k$. Then $(N_i - 1)/F_k \leqslant q_i \leqslant (N_i + 1)/F_k$. Let $f_i = N_i/F_k$. Then there exists $0 < \varepsilon < 1$ such that

$$1 - \varepsilon \leqslant \frac{N_i}{N_i + 1} \leqslant \frac{f_i}{q_i} \leqslant \frac{N_i}{N_i - 1} \leqslant 1 + \varepsilon.$$

For our construction we choose F_k large enough such that $(1 - \varepsilon) \leqslant f_i/q_i \leqslant (1 + \varepsilon)$ for a small ε as was described in [8]. For these $f_1, \ldots, f_m$ we construct the golden ratio sequence of length F_k, as defined in theorem 7.

Theorem 8. Let $C(\phi)$ be the cost of the Fibonacci schedule and let $C(F)$ be the cost of the optimal fractional schedule, under gap function $g(s) = s^{\alpha+1}$. Then for some $\nu > 0$,

$$C(\phi) < C(F) \cdot \nu^\alpha.$$

Proof. Let $q_1, \ldots, q_m$ be the optimal frequencies implied by the fractional model. Recall that $s_i = 1/q_i$. We get that the cost of the optimal schedule is $\sum_{i=1}^m p_i s_i^\alpha$ as stated in theorem 4. For the golden ratio sequence schedule, we bound the expected cost by $\sum_{i=1}^m p_i r_i^\alpha$, such that $\rho_i = r_i^\alpha / s_i^\alpha < \nu^\alpha$. Therefore,

$$\frac{\sum_{i=1}^m p_i r_i^\alpha}{\sum_{i=1}^m p_i s_i^\alpha} < \nu^\alpha.$$

Since all claims are valid for any r_i and s_i, for the simplicity we omit the index i from all variables. As a result, we refer to p_i, r_i, s_i, j_i, and S_i as p, r, s, j, and S, respectively. Our purpose is to find the smallest ν such that $(r/s) < \nu^\alpha$.

Since a gap of size x contributes $x^{\alpha+1}$ to the coefficient of p in calculating the cost, it follows that by the golden ratio sequence properties (theorem 7) we have

$$r^\alpha = \frac{1}{F_k}\left[S F_j^{\alpha+1} + (F_{k-j-2} + S) F_{j+1}^{\alpha+1} + (F_{k-j-1} - S) F_{j+2}^{\alpha+1}\right].$$

For the optimal solution we have,

$$s^\alpha = \left(\frac{F_k}{F_{k-j} + S}\right)^\alpha.$$

Hence the value for ρ is

$$\frac{1}{F_k}\left[S F_j^{\alpha+1} + (F_{k-j-2} + S) F_{j+1}^{\alpha+1} + (F_{k-j-1} - S) F_{j+2}^{\alpha+1}\right] \times \left(\frac{F_{k-j} + S}{F_k}\right)^\alpha.$$

By rearranging terms, the value of ρ is

$$\rho = \left(\frac{F_{k-j} + S}{F_k}\right)^\alpha \left[\frac{S F_j}{F_k} F_j^\alpha + \frac{(F_{k-j-2} + S) F_{j+1}}{F_k} F_{j+1}^\alpha + \frac{(F_{k-j-1} - S) F_{j+2}}{F_k} F_{j+2}^\alpha\right]. \quad (11)$$

By definition $j \leqslant k-2$. We proceed by case analysis. First, consider the case $j = k-2$. In this case, $S = 0$, i.e. i appears exactly once in the sequence, and using equation (11), we get

$$\begin{aligned}\rho &= \frac{F_{k-j}^\alpha (F_{k-j-2} F_{j+1}^{\alpha+1} + F_{k-j-1} F_{j+2}^{\alpha+1})}{F_k^{\alpha+1}} \\ &= \frac{F_2^\alpha (F_0 F_{k-1}^{\alpha+1} + F_1 F_k^{\alpha+1})}{F_k^{\alpha+1}} \\ &= \frac{F_k^{\alpha+1}}{F_k^{\alpha+1}} \\ &= 1.\end{aligned}$$

From now on we assume that $j \leqslant k-3$. Denote $\nu = \rho^{1/\alpha}$ (it will be more convenient to work with ν). Our goal now is to bound ν from above. First, note that equation (11) can be re-written as follows:

$$\nu = \frac{F_{k-j} + S}{F_k}\left[\frac{S F_j}{F_k} F_j^\alpha + \frac{(F_{k-j-2} + S) F_{j+1}}{F_k} F_{j+1}^\alpha + \frac{(F_{k-j-1} - S) F_{j+2}}{F_k} F_{j+2}^\alpha\right]^{1/\alpha}. \quad (12)$$

We can rewrite equation (12) ν as follows:

$$\nu = k_0\left[k_1 F_j^\alpha + k_2 F_{j+1}^\alpha + k_3 F_{j+2}^\alpha\right]^{1/\alpha}. \quad (13)$$

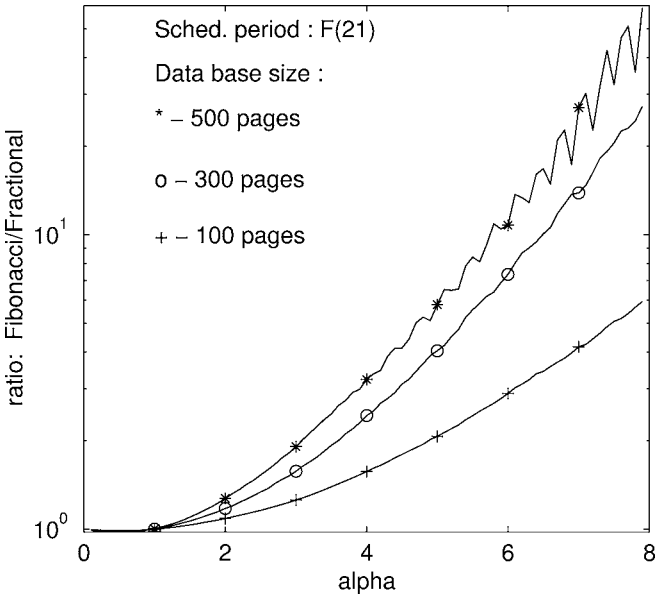

Figure 5. Performance of the Fibonacci algorithm as function of data base size.

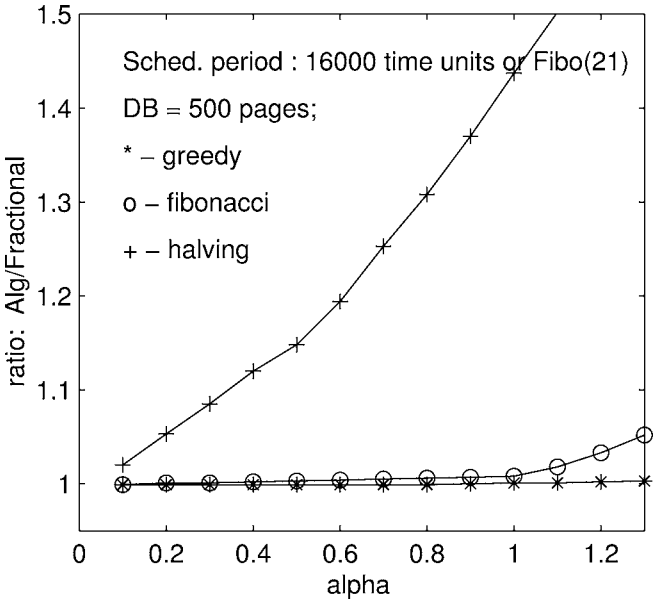

Figure 6. Performance of the discrete algorithms for sub-linear cost functions with a database of 500 pages.

Next, we evaluate the k_i coefficients in equation (13) using lemma 6 for $1 \leqslant j < k-2$ and $0 \leqslant S < F_{k-j-1}$:

$$k_0 = \frac{F_{k-j}+S}{F_k} < \frac{F_{k-j}+F_{k-j-1}}{F_k} = \frac{F_{k-j+1}}{F_k} < 1;$$

$$k_1 = \frac{SF_j}{F_k} < \frac{F_{k-j-1}F_j}{F_k} < \frac{F_{k-1}}{F_k} < 1;$$

$$k_2 = \frac{(F_{k-j-2}+S)F_{j+1}}{F_k} < \frac{(F_{k-j-2}+F_{k-j-1})F_{j+1}}{F_k}$$
$$= \frac{F_{k-j}F_{j+1}}{F_k} < \frac{F_{k+1}}{F_k} \leqslant 2;$$

$$k_3 = \frac{(F_{k-j-1}-S)F_{j+2}}{F_k} < \frac{F_{k-j-1}F_{j+2}}{F_k} < \frac{F_{k+1}}{F_k} \leqslant 2.$$

Hence, from equation (13) (and using lemma 6 once again) we get

$$\begin{aligned}\nu &< \frac{F_{k-j}+S}{F_k}\left[F_j^\alpha + 2F_{j+1}^\alpha + 2F_{j+2}^\alpha\right]^{1/\alpha} \\ &< \frac{F_{k-j}+S}{F_k}\left[4F_{j+2}^\alpha\right]^{1/\alpha} \\ &< \frac{F_{k-j+1}F_{j+2}}{F_k}\cdot 4^{1/\alpha} \\ &< \frac{F_{k+3}}{F_k}4^{1/\alpha} \leqslant 5\cdot 4^{1/\alpha}.\end{aligned}$$

Which means that ν is finite for all $\alpha > 0$. □

Since we evaluated the upper bound for ρ numerically, it was easier to evaluate the small numbers involved in computing ν rather than the large numbers involved in computing ρ.

We estimate that $\nu \approx 1.9$. Our experiments support the theoretical and numerical analysis. By the experiments whose results are plotted in figure 5 we arrive at the following conclusion. Specifically, let ϕ_{500} be a typical schedule generated by the Fibonacci algorithm in a 500-page database. Then

$$C(\phi_{500}) \approx 0.50 \cdot 1.70^\alpha. \tag{14}$$

Compared to the Halving algorithm, the Fibonacci algorithm produces much better results for small databases. The results of the Fibonacci algorithm are also remarkably close to optimum in the case of small α values when the database is large (see figure 6). It is interesting to note in figure 5 the phenomenon of non-monotonic behavior: sometimes, the cost of the algorithm *drops* when α increases.

5.4. The Greedy algorithm

The Greedy algorithm is very simple and intuitively appealing. At each time step, the Greedy algorithm broadcasts the page which will incur the largest cost if not broadcasted. Note that as stated, the Greedy algorithm requires non-trivial computation at each step. To circumvent this difficulty, we compute a finite schedule using the Greedy algorithm, and then repeat it. We have experimented with the Greedy schedule to test the effect of the size of the database and the power of the cost function. We discovered that while still exponential in the degree of the cost polynomial, the performance of the Greedy schedule is extremely close to the cost of the optimal fractional schedule. As can be seen in figure 7, the Greedy schedule has incurs high cost relative to the cost of a fractional schedule only when the database is large and when cost polynomial is of high degree. Quantitatively, if we denote by G_{500} a typical schedule generated by the Greedy algorithm for a 500 page database, then by our measurements we have that

$$C(G_{500}) \approx 0.07 \cdot 1.44^\alpha. \tag{15}$$

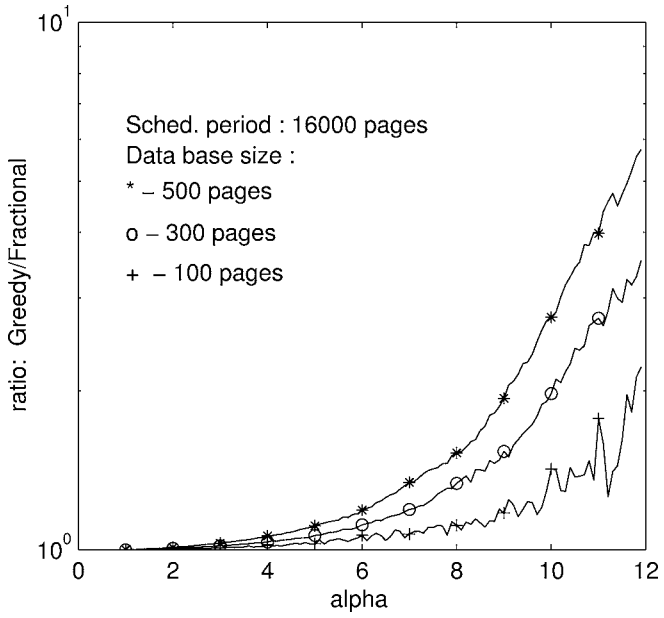

Figure 7. Performance of the Greedy algorithm as a function of the database size (note the scale).

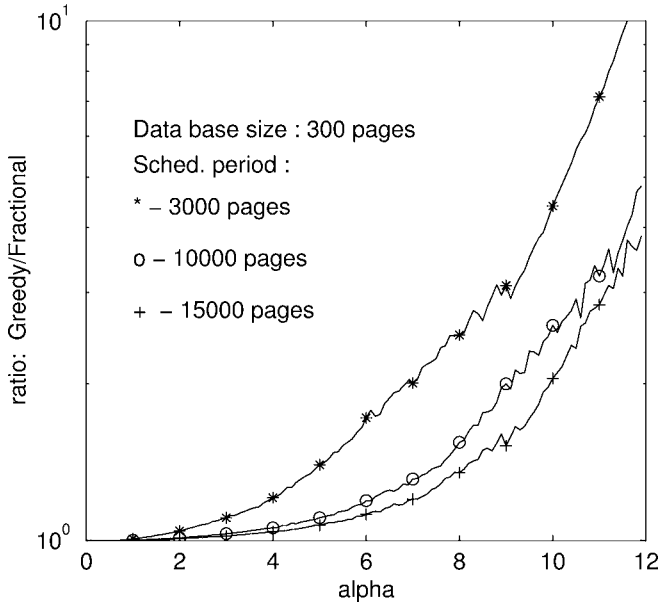

Figure 8. Performance of the Greedy algorithm as a function of the cutoff point.

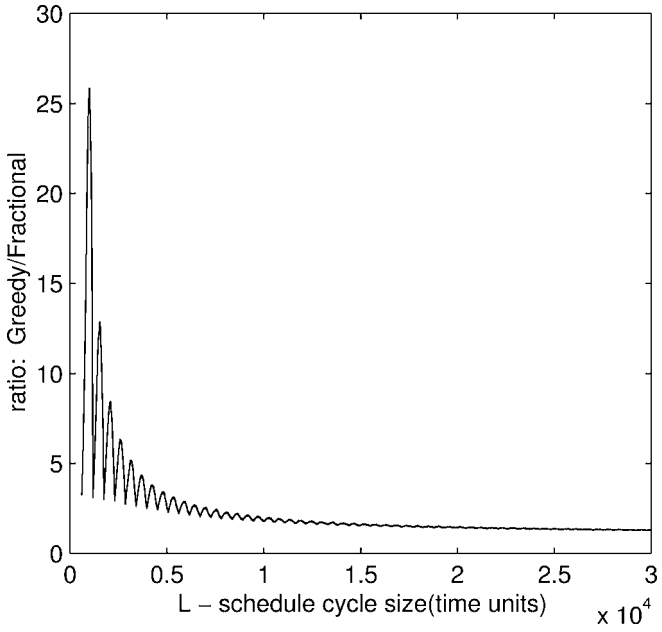

Figure 9. Performance of the Greedy algorithm as a function of the cutoff point with cost function $c(t) = t^8$ and database size 500 pages.

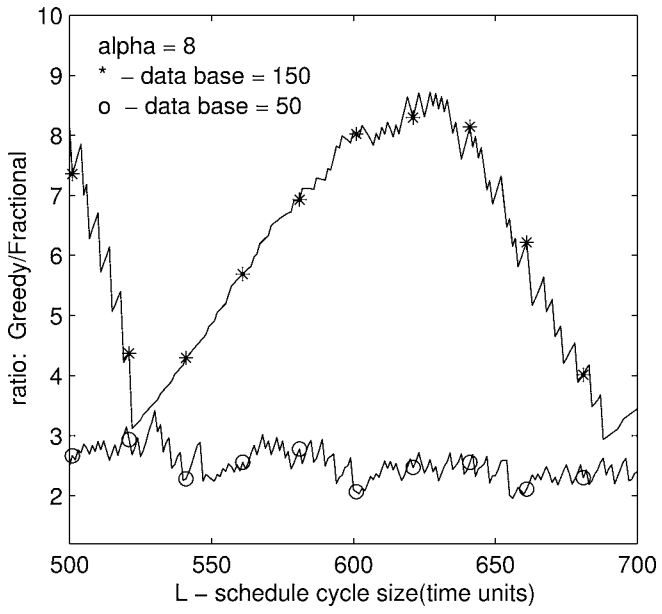

Figure 10. Performance of the Greedy algorithm as a function of the cutoff point with for different database sizes.

To gain further insight into the performance of Greedy schedules, we measured its cost further, now as a function of the cycle cutoff point. As expected, generally speaking, the larger is the cycle length, the better results the schedule yields (figure 8).

It seems that a reasonable cutoff point for the Greedy schedule is about $10\alpha m$, where α is the degree of the cost polynomial, and m is the size of the database. A closer inspection of the expected cost as a function of the cutoff point, showed, to our surprise, that the cost is *not* a monotonic function of the schedule cycle length. A dramatic example is depicted in figure 9, where the initial fluctuations may change the expected cost by a factor as high as 10. More generally, we can conclude that the cost function behaves periodically, as is easy to see in figure 10. The period length is (quite closely) the size of the database, independent of α. Therefore, it seems that when constructing a finite approximation of the Greedy schedule, one should find the least costly point in the range $[10\alpha m, 10\alpha m + m]$.

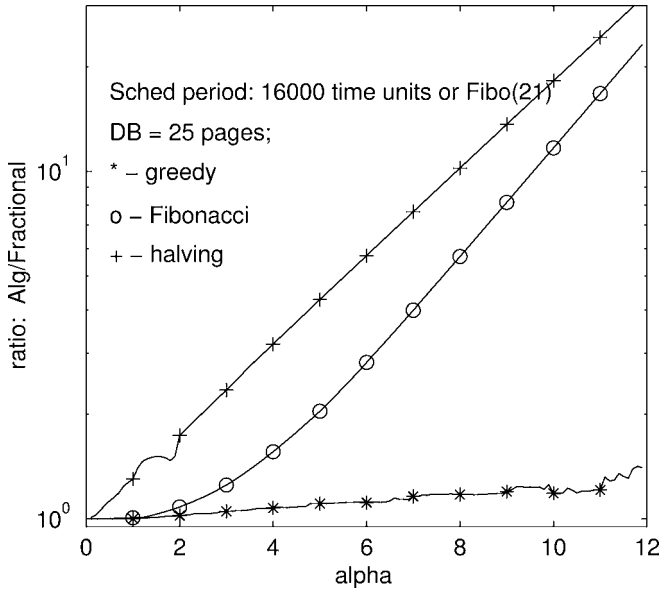

Figure 11. Comparison of the performance of the discrete algorithms as a function of α for a small database (Random does not fit in the scale).

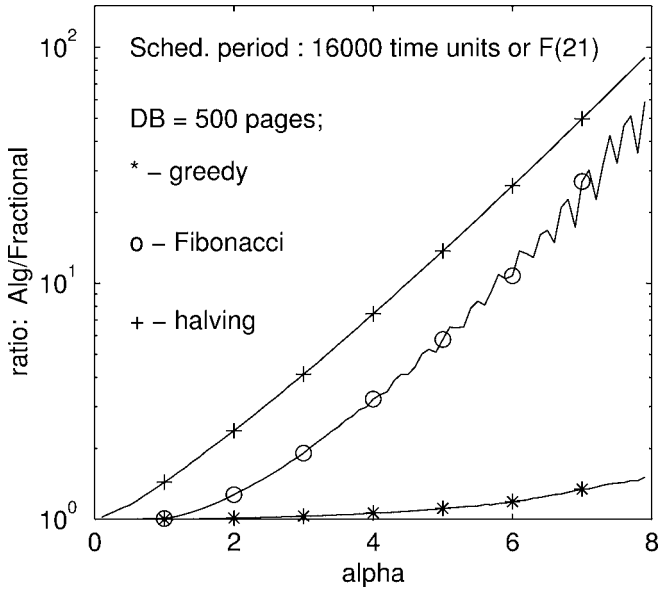

Figure 12. Comparison of the performance of the discrete algorithms as a function of α for a large database (Random does not fit in the scale).

6. Conclusion

In this paper we have proposed and analyzed polynomial cost measure for broadcast disk system: most previous work was done for linear cost functions. We have given an algorithm for asymptotically optimal schedules in the fractional model, where the period of each item may be fractional. We have compared four popular algorithms which try to approximate the fractional schedules. We found that the behavior of these algorithms is exponential in the degree of the cost polynomial, both analytically and experimentally. Intuitively, this phenomenon stems from the fact that for all these algorithms, from time to time there is a page which is served much less frequently than it should according to the fractional solution. When the power of the cost polynomial is high, any such deviation has large impact on the total cost.

We discovered substantial differences between these algorithms, however. The best algorithm was – usually by far – the Greedy algorithm. A summary of these results is given in figures 11 and 12. For sub-linear cost functions, we found that the Fibonacci algorithm also enjoys excellent performance (figure 6). We have studied the behavior of the Greedy algorithm, and found that its performance depends critically (for large databases and for cost functions which are high-degree polynomials) on the length of the cycle taken. We were able to determine specific values for the cycle.

References

[1] S. Acharya, R. Alonso, M.J. Franklin and S. Zdonik, Broadcast disks: data management for asymmetric communications environment, in: *Proc. of ACM SIGMOD Int. Conference on Management of Data (SIGMOD'95)*, San Jose, CA (June 1995) pp. 199–210.

[2] S. Acharya, M.J. Franklin and S. Zdonik, Dissemination-based data delivery using broadcast disks, IEEE Personal Communications 2(6) (1995) 50–60.

[3] M.H. Ammar and J.W. Wong, The design of teletext broadcast cycles, Performance Evaluation 5(4) (1985) 235–242.

[4] M.H. Ammar and J.W. Wong, On the optimality of cyclic transmission in teletext systems, IEEE Transactions on Communication 35(1) (1987) 68–73.

[5] S. Anily and J. Bramel, Periodic scheduling with service constraints, Manuscript (1997).

[6] S. Anily, C.A. Glass and R. Hassin, The scheduling of maintenance service, Discrete Applied Mathematics 82 (1998) 27–42.

[7] S. Anily, C.A. Glass and R. Hassin, Scheduling maintenance service to three machines, Annals of Operation Research 86 (1999) 375–391.

[8] A. Bar-Noy, R. Bhatia, J. Naor and B. Schieber, Minimizing service and operation costs of periodic scheduling, Mathematics of Operations Research 27(3) (August 2002) 518–544.

[9] A. Bar-Noy and Y. Shilo, Optimal broadcasting of two files over an asymmetric channel, Journal of Parallel and Distributed Computing 60(4) (April 2000) 474–493.

[10] L. Breslau, P. Cao, L. Fan, G. Phillips and S. Shenker, Web caching and Zipf-like distributions: evidence and implications, in: *Proc. of the 18th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'99)* (1999).

[11] M.Y. Chan and F. Chin, Schedulers for larger classes of pinwheel instances, Algorithmica 9 (1993) 425–462.

[12] M.Y. Chan and F. Chin, General schedulers for the pinwheel problem based on double-integer reduction, IEEE Transactions on Computers 41(6) (June 1992) 755–768.

[13] C.A. Glass, Feasibility of scheduling lot sizes of three products on one machine, Management Science 38 (1992) 1482–1494.

[14] C.A. Glass, Feasibility of scheduling lot sizes of two frequencies on one machine, European Journal of Operation Research 75 (1994) 354–364.

[15] R. Hassin and N. Megiddo, Exact computation of optimal inventory policies over an unbounded horizon, Mathematics of Operations Research 16 (1991) 534–546.

[16] M. Hofri and Z. Rosberg, Packet delay under the golden ratio weighted TDM policy in a multiple-access channel, IEEE Transactions on Information Theory 11-33 (1987) 341–349.

[17] R. Holte, L. Rosier, I. Tulchinsky and D. Varvel, Pinwheel scheduling with two distinct numbers, Theoretical Computer Science 100 (1992) 105–135.

[18] T. Imielinski, S. Viswanathan and B. Badrinath, Energy efficient index-

ing on air, in: *Proc. of ACM SIGMOD International Conference on Management of Data (SIGMOD'94)* (1994) pp. 25–36.
[19] A. Itai and Z. Rosberg, A golden ratio control policy for a multiple-access channel, IEEE Transactions on Automatic Control 29 (1984) 712–718.
[20] C. Kenyon and N. Schabanel, The data broadcast problem with non-uniform transmission times, in: *Proc. of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'99)* (1999) pp. 547–556.
[21] D.E. Knuth, *The Art of Computer Programming*, Vol. 1 (Addison-Wesley, Reading, MA, 1973).
[22] Z. Rosberg, Cell multiplexing in ATM networks, IEEE Transactions on Networking 4 (February 1996) 112–122.
[23] R. Roundy, 98%-effective integer-ratio lot-sizing for one-warehouse multi-retailer systems, Management Science 31 (1985) 1416–1430.
[24] C.J. Su and L. Tassiulas, Broadcast Scheduling for Information Distribution, in: *Proc. of the 16th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'97)* (1997) pp. 109–117.
[25] N. Vaidya and S. Hameed, Log time algorithms for scheduling single and multiple channel data broadcast, in: *Proc. of the 3rd Annual ACM/IEEE International Conference on Mobile Computing and Networking* (1997) pp. 90–99.
[26] N. Vaidya and S. Hameed, Data broadcast scheduling: On-line and off-line algorithms, Technical report 96-017, Computer Science, Texas A&M University (July 1996).
[27] W. Wei and C. Liu, On a periodic maintenance problem, Operations Research Letters 2 (1983) 90–93.
[28] J.W. Wong, Broadcast delivery, Proceedings of IEEE (December 1988) 1566–1577.

Amotz Bar-Noy received the B.Sc. degree in 1981 in mathematics and computer science and the Ph.D. degree in 1987 in computer science, both from the Hebrew University, Israel. From October 1987 to September 1989 he was a post-doc fellow in Stanford University, California. From October 1989 to August 1996 he was a Research Staff Member with IBM T.J. Watson Research Center, New York. From February 1995 to September 2001 he was an Associate Professor with the Electrical Engineering-Systems Department of Tel Aviv University, Israel. From September 1999 to December 2001 he was with AT&T Research Labs in New Jersey. Since February 2002 he is a Professor with the Computer and Information Science Department of Brooklyn College, New York. His main interests are: networks and graph algorithms, mobile and wireless networks algorithms, scheduling, and combinatorial optimization.
E-mail: amotz@sci.brooklyn.cuny.edu

Boaz Patt-Shamir has received his BSc in mathematics and computer science from Tel Aviv University, his MSc in computer science from Weizmann Institute, and his PhD in computer science from MIT. Following that he was on the faculty of Northeastern University. Since 1997 he is with the Department of Electrical Engineering in Tel Aviv University, where he directs the laboratory for computer networks and multimedia. His main research interests include distributed systems and network algorithms.
E-mail: boaz@eng.tau.ac.il

Igor Ziper has received his BSc in electrical and computer engineering from Ben-Gurion University (Beer-Sheva), his MSc in electrical and computer engineering from Tel Aviv University. Since 2000 he is working for Integrated Device Technology (IDT), where he develops the new generation of chips for different network applications. His main research interests include network protocols and systems.
E-mail: igor@eng.tau.ac.il