

Broadcast Disks with Polynomial Cost Functions

Amotz Bar-Noy[†]

amotz@research.att.com

AT&T Research Labs

180 Park Ave.

Florham Park, NJ 07932

Boaz Patt-Shamir

boaz@eng.tau.ac.il

Dept. of Electrical Engineering

Tel Aviv University

Tel Aviv 69978, Israel

Igor Ziper

igor@eng.tau.ac.il

Abstract—In broadcast disk systems, information is broadcasted in a shared medium. When a client needs an item from the disk, it waits until that item is broadcasted. 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 fluid 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.

I. INTRODUCTION

THE idea of *broadcast disks*, introduced by Franklin *et al.* [3], [1], has gained considerable popularity lately (see e.g., [8], [9], [18], [20], [24], [27], [28]). 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, as is usually the case with mobile hosts and with Teletext system [6], [7].

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 be solved by using faster shared media, but in relative terms, better solutions are required. Indeed, it was shown that some a-priori 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 where $c(i) = i$, 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 be 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 1 (sub-linearity).

Our contributions.

First, we analyze the known “fluid” relaxations to a general polynomial cost functions. The idea behind the fluid relaxation is to consider the problem where the bandwidth can be partitioned with arbitrary precision among any number of concurrent streams, where each stream contains a single page. The objective then becomes to find the best way to allocate bandwidth to pages. Obviously, a discrete schedule (which is our main interest) can be translated into a fluid relaxation by setting the bandwidth share to be the reciprocal of the average frequency in the discrete model. The converse, however, does not hold in general. We show how to solve the fluid relaxations asymptotically optimally for any polynomial cost function. The fluid 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 fractional bandwidths for the fluid model.

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

[†] On leave from Dept. of Electrical Engineering, Tel Aviv University.

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

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 II we give a brief description of related work. Section III describes the model and introduces some terminology. Section IV explains how to find the lower bound for the overall mean cost for general polynomial cost functions via the fluid model. We remark that all proofs are omitted from this version of the paper. Section V presents the evaluation of the various algorithms for polynomial cost functions.

II. 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 [6], [7] 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 the mean response time obtained by a fractional solution [6], [7]. The greedy rule that is used by our algorithm is similar to the one in the heuristic of [4] and is the same as the *mean aggregate delay* rule described in [24].

Anily *et al.* [4] consider the problem in the setting of the maintenance problem, where the cost is linear in the waiting time. In [5], 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 [2]. 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], [12], [13], [14], [15], [17], [23], [25].

III. 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.

A. Basic Model (Discrete)

The *server database* contains m information items called *pages*, denoted by A_1, A_2, \dots, 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 $S = S_0, S_1, S_2, \dots, S_{N-1}$, where $S_t \in \{A_1, A_2, \dots, A_m\}$ for all t . Henceforth, we consider only cyclic schedules (this is justified by the result of Anily and Bramel [2] that there always exists an optimal cyclic schedule). The letter N will be 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 $t + 1, \dots, t + x - 1$. 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 completely 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 .

B. 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 :

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

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

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

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.

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

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

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.

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

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 definition for equal-spacing schedules, where the space between successive instances of A_i is s_i .

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

IV. ANALYSIS OF THE FLUID MODEL

In this section we consider a different variant of the model presented in Section III. This model, called the *fluid model* (or *fractional model*), is both interesting in its own right and also helpful in lower-bounding the cost of schedules in our main (discrete) model. It also serves as a starting point of some of the algorithms we study.

The fluid model is defined as follows. We are given a database with demand probabilities as before. Here, however, time is continuous, and in each time instant we are allowed to allocate *fractions* of the total bandwidth to different pages, so long as the total allocated bandwidth never exceeds one unit. (Intuitively, one may think that we substitute discrete time division multiplexing with continuous frequency division multiplexing.) A customer wishing to read a page must wait until a complete unit of the requested page is broadcasted. Formally, a schedule is given by a set of m continuous, non-negative functions $b_i(t)$,

such that for all times t , we have that $\sum_{i=1}^m b_i(t) \leq 1$. For each $1 \leq i \leq m$, the function $b_i(t)$ represents the bandwidth share allocated to page A_i at time t . A customer arriving at time t_0 wishing to get page A_i must wait until the first time t_1 such that $\int_{t_0}^{t_1} b_i(t) dt = 1$.

The motivation for considering the fluid model is twofold. First, it can serve as a faithful abstraction of certain physical systems. Secondly, the fluid model is more convenient to analyze, which helps us in understanding the discrete model better. Specifically, in the fluid model, it is sufficient to determine only the *average frequency* of a given page, as implied by the following result. Recall that the average frequency of a page in a cyclic schedule with cycle length N is the number of its occurrences in the cycle divided by N .

Proposition 1: *Suppose that the average frequency of a page A_i in schedule must be some $0 \leq q_i \leq 1$. Then for any monotonically increasing cost function, the minimal cost incurred by a random customer waiting for A_i is achieved in a fluid schedule with $b_i(t) = q_i$ for all t .*

Note that one can try to convert a fluid schedule with fixed bandwidth allocations q_i to a discrete schedule by assigning a spacing of $1/q_i$ for page A_i . If this works, then optimality is guaranteed by Proposition 1 (with respect to the given q_i -s). The problem is that $1/q_i$ is not necessarily integral. (This is the reason why a solution in the fluid model is often called *fractional schedule*.) Furthermore, even if $1/q_i$ were integral for all i , it may be the case that no discrete schedule with spacing $1/q_i$ exists. (Consider, for example, $q_1 = 1/2$ and $q_2 = 1/3$; there is no way to schedule A_1 exactly every two time units and A_3 exactly every three time units.) Thus, the fluid model serves as a relaxation of the discrete model which enables us to isolate the problem of finding the optimal average frequencies. Clearly, any discrete schedule incurs at least as much cost as the optimal fluid schedule. In other words, the cost derived from a fractional solution to the fluid model is a lower bound on the cost in the discrete model. We use this observation by calculating ideal spacing values in the fluid 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).

A. Optimal Fluid Schedule for General Linear Cost

Suppose that the cost function is the general polynomial of degree 1, denoted $c(x) = \beta x + \gamma$. 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 :

$$\begin{aligned} C_i &= \frac{1}{s_i} \sum_{w=1}^{s_i} (\beta w + \gamma) \\ &= \beta \frac{s_i + 1}{2} + \gamma. \end{aligned}$$

Next, we calculate the overall expected cost.

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

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

Theorem 2: *In the fluid model, the minimum expected cost is achieved when the frequency q_i of each page A_i satisfies*

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

This result complements known results [7], [26], [27] about the ‘‘square root rule’’ for minimizing waiting times, i.e., for the identity cost function $c(t) = t$. The proof of Theorem 2 is similar to proof of ‘‘square root rule’’ theorem given in [7], and will be given in the full paper.

B. Asymptotically Optimal Solutions for General Polynomial Cost Functions

To deal with general polynomial cost functions, we switch to dealing with gap functions. We first prove basic connections between cost functions and gap functions.

The following combinatorial 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: First we represent $g(t-1)$ using the Binomial Law:

$$\begin{aligned} g(t-1) &= (t-1)^{\alpha+1} \\ &= (-1)^{\alpha+1} + \binom{\alpha+1}{1} (-1)^{\alpha} t \\ &\quad + \binom{\alpha+1}{2} (-1)^{\alpha-1} t^2 \\ &\quad \dots + \binom{\alpha+1}{\alpha+1} t^{\alpha+1} \\ &= \sum_{j=0}^{\alpha+1} (-1)^{\alpha+1-j} \binom{\alpha+1}{j} t^j. \end{aligned}$$

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

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

Using Lemma 3, we can find the cost function for any given polynomial gap function. Together with Eq. (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 $g(s_i) = s_i^{\alpha+1}$, then the optimal schedule is achieved when the spacing between any two successive instances of A_i is*

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

The proof of Theorem 4 is similar to proof of Theorem 2, and will be given in the full paper.

Using Theorem 4, we can find asymptotically optimal schedules for any polynomial gap functions.

Theorem 5: *If the cost function is $c(t) = t^{\alpha} + O(t^{\alpha-1})$ for some $\alpha > 0$, then the optimal schedule is achieved when the spacing s_i between any two successive instances of A_i satisfies*

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

The proof of Theorem 5 will be given in the full paper.

Thus we have a good asymptotic estimate for optimal schedules in the fluid 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.

V. 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., [27], [10]). 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 $1 \leq i \leq m$.

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

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 we 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].

A. The Random Algorithm

The Random algorithm tries to approximate the fluid schedule by using its fractional spacings. Specifically, let $\{q_i\}$ be the frequencies of pages in the optimal fluid 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

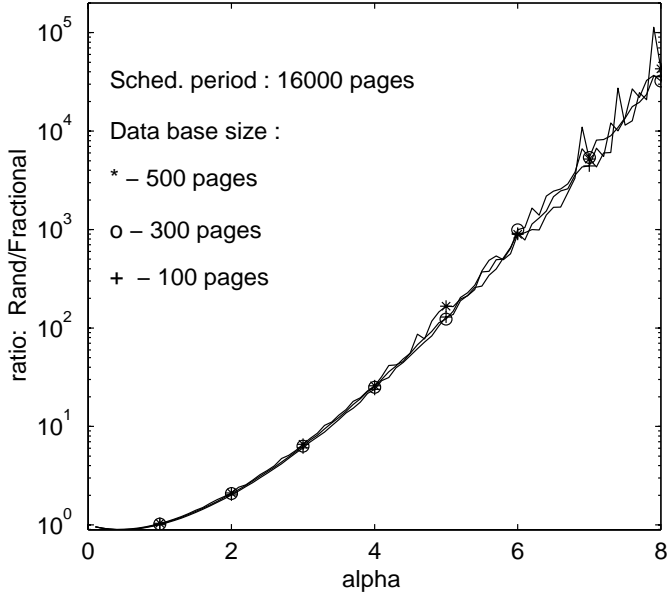


Fig. 1. Performance of Random algorithm as function of data base size

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 fluid 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 $(\frac{\alpha+1}{e})^{\alpha+1}$) times worse than the fluid 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

$$C(R_{500}) \approx 0.02 \cdot 6.12^\alpha. \quad (8)$$

In other words, even for randomly generated databases, the cost for schedules generated by the Random algorithm is exponentially more expensive than the fluid schedules. This result is independent of the database size (Figure 1) and is already evident in relatively short schedules (Figure 2).

B. 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, 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

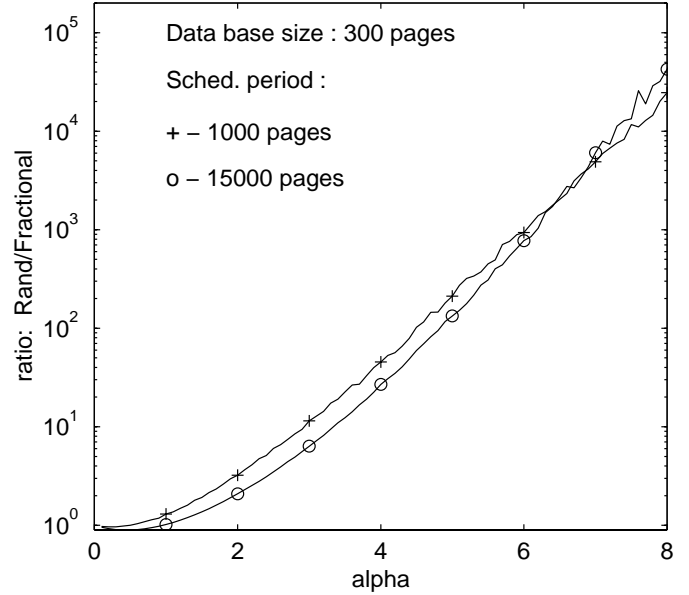


Fig. 2. Performance of Random algorithm as function of schedule size

guaranteed performance is always less than twice the cost of the optimal fluid 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 fluid model. The cost with fractional spacing obtained from the optimal fluid schedule S_F is

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

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

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

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 3). 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} .

$$C(H_{500}) \approx 0.76 \cdot 1.80^\alpha. \quad (9)$$

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.

C. The Fibonacci (Golden Ratio) Algorithm

The Fibonacci algorithm generates a schedule with the same average frequencies as those of the optimal fluid solution. However, unlike the fluid model, the spacing between two consecutive appearances of the same page in the schedule may have

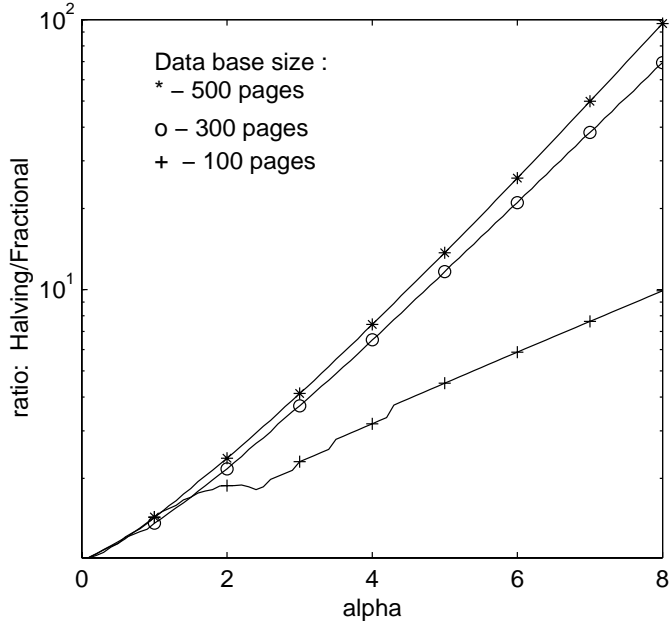


Fig. 3. Performance of the Halving algorithm as a function of the database size

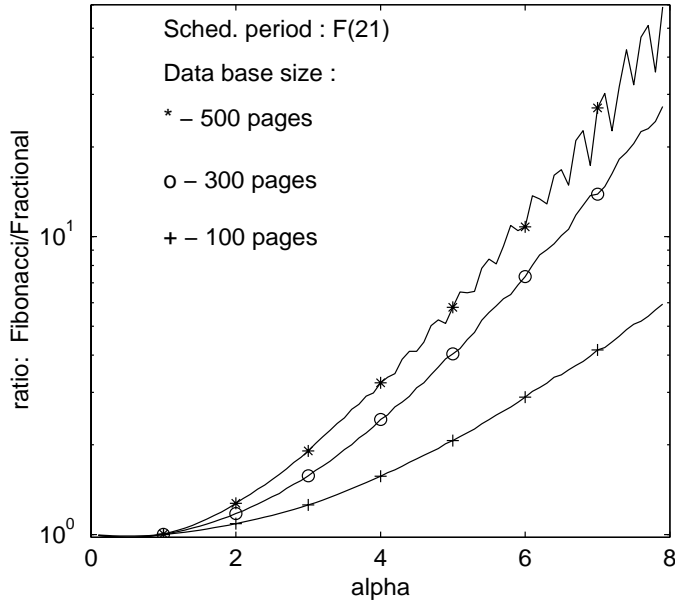


Fig. 4. Performance of the Fibonacci algorithm as function of data base size

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 fluid-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 inter-

ested 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 $k > 0$, $\frac{F_{k+1}}{F_k} \leq 2$.
- For all $k > 0$, $\frac{F_{k+3}}{F_k} \leq 5$.

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, \dots, g_m be frequencies such that $\sum_{i=1}^m g_i = 1$, and let F_k be a Fibonacci number such that $g_i = \frac{N_i}{F_k} = \frac{F_{k-j_i} + S_i}{F_k}$ for some $1 \leq j_i \leq k-2$ and $0 \leq S_i < F_{k-j_i-1}$ ($\sum_{i=1}^m N_i = F_k$). Then there exists a sequence of length F_k of the pages A_1, \dots, A_m such that A_i appears N_i times in the sequence. Furthermore, for each $1 \leq i \leq m$, 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 proof in [8]).

The golden ratio sequence: Let q_1, \dots, q_m denote frequencies ($q_i = 1/s_i$, where s_1, \dots, 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 $\frac{N_i-1}{F_k} \leq q_i \leq \frac{N_i+1}{F_k}$. Let $f_i = N_i/F_k$. Then there exists $0 < \varepsilon < 1$ such that

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

For our construction we choose F_k large enough such that $(1 - \varepsilon) \leq \frac{f_i}{q_i} \leq (1 + \varepsilon)$ for a small ε as was described in [8]. For these f_1, \dots, 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, \dots, q_m be the optimal frequencies implied by the fluid 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 $\rho_i = \frac{r_i^\alpha}{s_i^\alpha} < \nu^\alpha$. 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,

$$\begin{aligned} \rho &= \frac{1}{F_k} \left[S F_j^{\alpha+1} + (F_{k-j-2} + S) F_{j+1}^{\alpha+1} + (F_{k-j-1} - S) F_{j+2}^{\alpha+1} \right] \left(\frac{F_{k-j} + S}{F_k} \right)^\alpha \\ &= \left(\frac{F_{k-j} + S}{F_k} \right)^\alpha \cdot \left[\frac{S F_j}{F_k} F_j^\alpha + \frac{(F_{k-j-2} + S) F_{j+1}}{F_k} F_{j+1}^\alpha + \frac{(F_{k-j-1} - S) F_{j+2}}{F_k} F_{j+2}^\alpha \right]. \end{aligned} \quad (10)$$

By definition $j \leq k-2$. 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 Eq. (10), 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 $j \leq k-3$. 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 Eq. (10) can be re-written as follows.

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

We can rewrite Eq. (11) ν as follows.

$$\nu = k_0 [k_1 F_j^\alpha + k_2 F_{j+1}^\alpha + k_3 F_{j+2}^\alpha]^{\frac{1}{\alpha}}. \quad (12)$$

Next, we evaluate the k_i coefficients in Eq. (12) using Lemma 6 for $1 \leq j < k-2$ and $0 \leq S < F_{k-j-1}$.

$$\begin{aligned} k_0 &= \frac{F_{k-j} + S}{F_k} \\ &< \frac{F_{k-j} + F_{k-j-1}}{F_k} = \frac{F_{k-j+1}}{F_k} < 1; \\ k_1 &= \frac{S F_j}{F_k} < \frac{F_{k-j-1} F_j}{F_k} < \frac{F_{k-1}}{F_k} < 1; \\ k_2 &= \frac{(F_{k-j-2} + S) F_{j+1}}{F_k} \\ &< \frac{(F_{k-j-2} + F_{k-j-1}) F_{j+1}}{F_k} \\ &= \frac{F_{k-j} F_{j+1}}{F_k} < \frac{F_{k+1}}{F_k} \leq 2; \\ k_3 &= \frac{(F_{k-j-1} - S) F_{j+2}}{F_k} \\ &< \frac{F_{k-j-1} F_{j+2}}{F_k} < \frac{F_{k+1}}{F_k} \leq 2; \end{aligned}$$

Hence, from Eq. (12) (and using Lemma 6 once again) we get

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

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 4 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

$$C(\phi_{500}) \approx 0.50 \cdot 1.70^\alpha. \quad (13)$$

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 5). It is interesting to note in Figure 4 the phenomenon of non-monotonic behavior: sometimes, the cost of the algorithm *drops* when α increases.

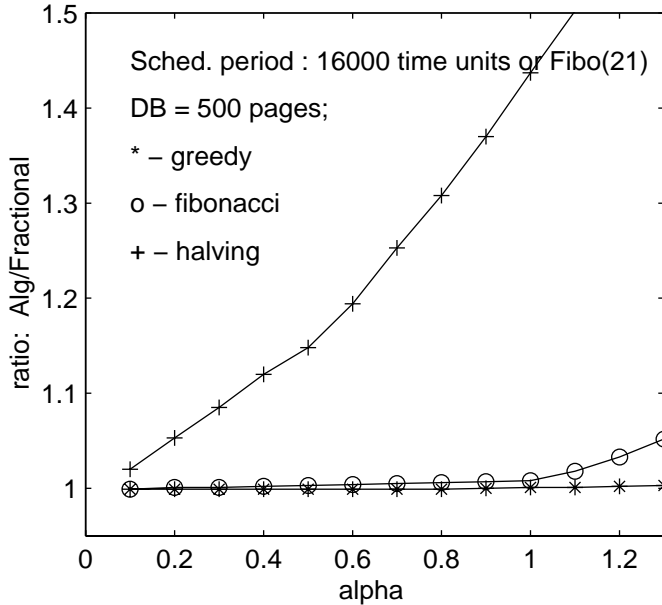


Fig. 5. Performance of the discrete algorithms for sublinear cost functions with a database of 500 pages

D. 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 6, the Greedy schedule has incurred 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

$$C(G_{500}) \approx 0.07 \cdot 1.44^\alpha. \quad (14)$$

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 7).

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 8, 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 9. The period length is (quite closely) the size of the database,

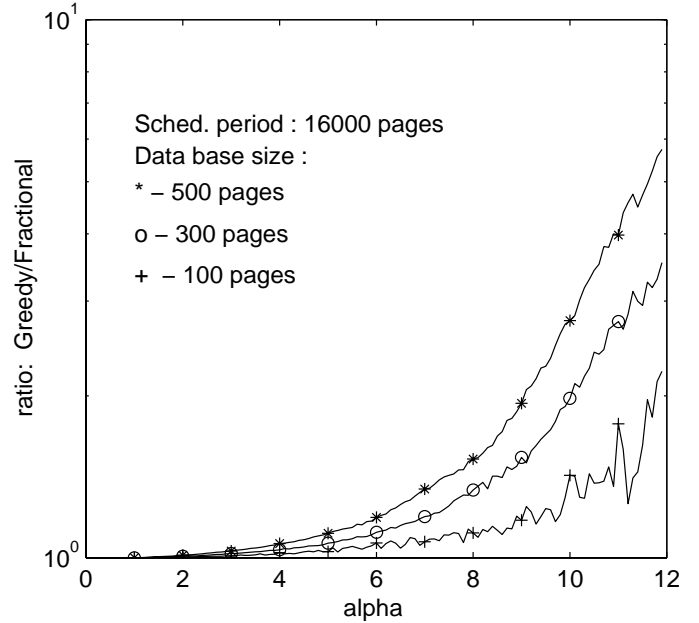


Fig. 6. Performance of the Greedy algorithm as a function of the database size (note the scale)

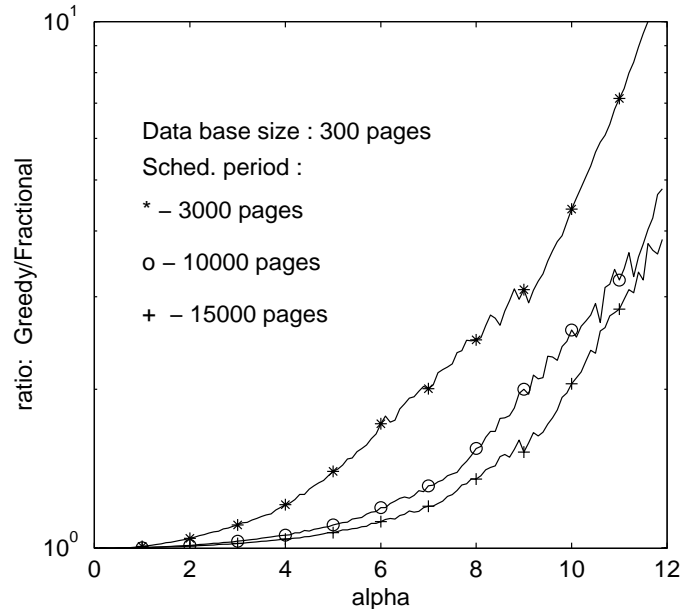


Fig. 7. Performance of the Greedy algorithm as a function of the cutoff point

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 $[10\alpha m, 10\alpha m + m]$.

VI. 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 fluid 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 ex-

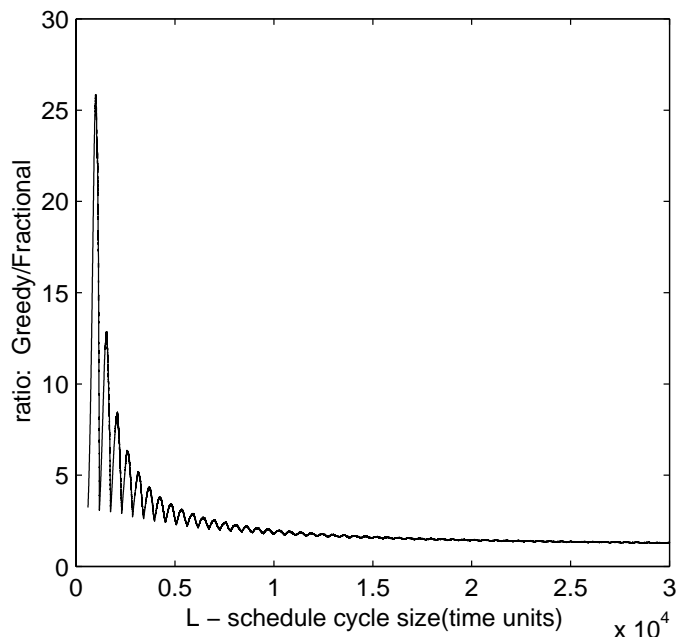


Fig. 8. Performance of the Greedy algorithm as a function of the cutoff point with cost function $c(t) = t^8$ and database size 500 pages

potential 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 10 and 11. For sub-linear cost functions, we found that the Fibonacci algorithm also enjoys excellent performance (Figure 5). 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*. ACM SIGMOD Int. Conference on Management of Data (SIGMOD 95), San Jose, CA, 199-210, June 1995.
- [2] S. Anily and J. Bramel. *Periodic Scheduling with Service Constraints*. Manuscript, 1997.
- [3] S. Acharya, M. J. Franklin, and S. Zdonik. *Dissemination-based Data Delivery Using Broadcast Disks*. IEEE Personal Communications, Vol. 2(6), 50–60, 1995.
- [4] S. Anily, C. A. Glass and R. Hassin. *The Sceduling of Maintenance Service*. Discrete Applied Mathematics, Vol. 82(1998), 27–42, 1998.
- [5] S. Anily, C. A. Glass and R. Hassin. *Sceduling Maintenance Service to Three Machines*. Annals of Operation Research, Vol. 86(1999), 375–391, 1999.
- [6] M. H. Ammar and J. W.Wong. *The Design of Teletext broadcast cycles*. Performance Evaluation, Vol. 5(4), 235–242, 1985.
- [7] M. H. Ammar and J. W.Wong. *On the optimality of cyclic transmission in Teletext systems*. IEEE Transaction on Communication, COM-35(1), 68–73, 1987.

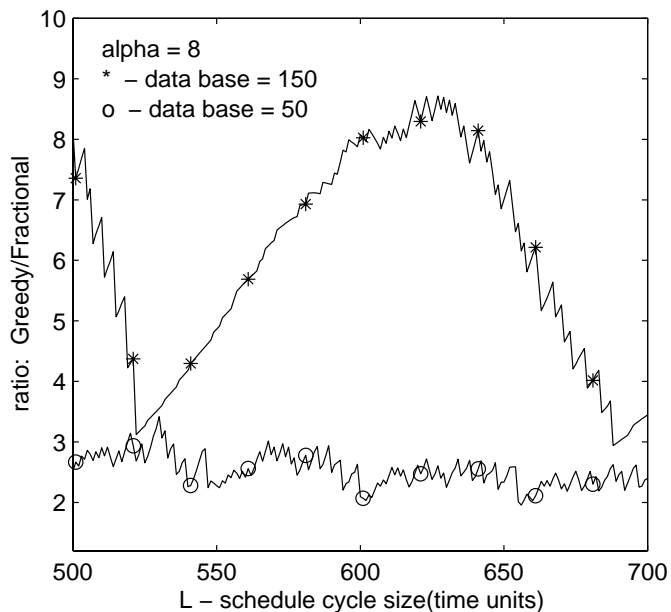


Fig. 9. Performance of the Greedy algorithm as a function of the cutoff point with for different database sizes

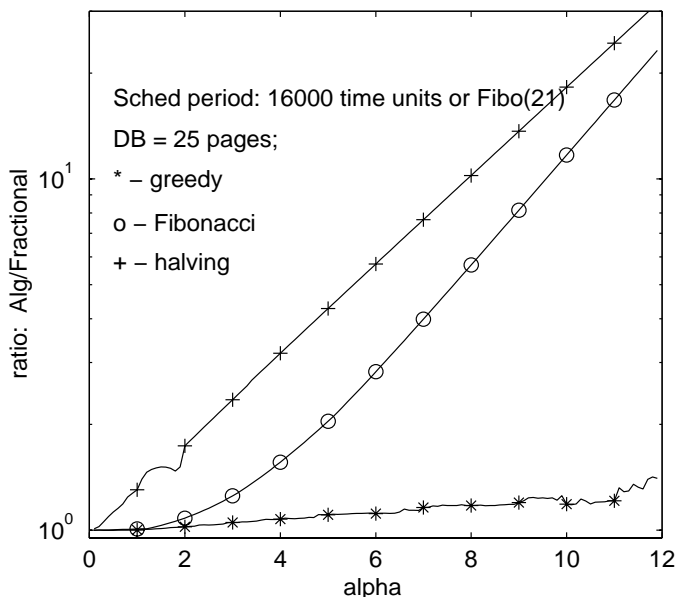


Fig. 10. Comparison of the performance of the discrete algorithms as a function of α for a small database (Random does not fit in the scale)

- [8] A. Bar-Noy, R. Bhatia, J. Naor, and B. Schieber. *Minimizing Service and Operation Costs of Periodic Scheduling*. Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 98), 11–20, 1998.
- [9] A. Bar-Noy and Y. Shilo. *Optimal Broadcasting of Two Files over an Asymmetric Channel*. Proceedings of the 18th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 99), 267–274, 1999.
- [10] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. *Web Caching and Zipf-like Distributions: Evidence and Implications*. Proceedings 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, Vol. 9, 425-462, 1993.
- [12] M. Y. Chan and F. Chin. *General schedulers for the pinwheel problem based on double-integer reduction*. IEEE Transactions on Computers, Vol.

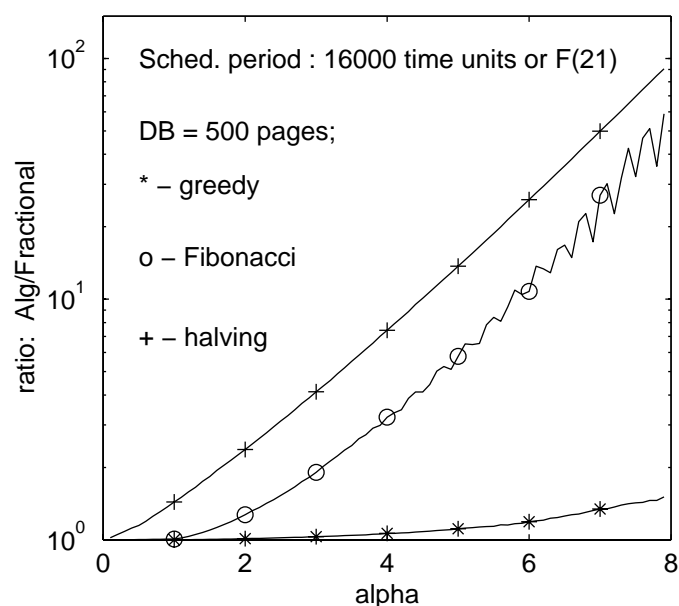


Fig. 11. Comparison of the performance of the discrete algorithms as a function of α for a large database (Random does not fit in the scale)

- 41(6), pp 755-768, June 1992.
- [13] C. A. Glass. *Feasibility of scheduling lot sizes of two frequencies on one machine*. European Journal of Operation Research 75, 354-364, 1994.
- [14] C. A. Glass. *Feasibility of scheduling lot sizes of three products on one machine*. Management Science 38, 1482-1494, 1992.
- [15] R. Hassin and N. Megiddo. *Exact computation of optimal inventory policies over an unbounded horizon*. Mathematics of Operations Research 16, 534-546, 1991.
- [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, Vol. 11-33, 341-349, 1987.
- [17] R. Holte, L. Rosier, I. Tulchinsky, and D. Varvel. *Pinwheel scheduling with two distinct numbers*. Theoretical Computer Science 100, 105-135, 1992.
- [18] T. Imielinski, S. Viswanathan, and B. Badrinath. *Energy efficient indexing on air*. ACM SIGMOD International Conference on Management of Data (SIGMOD 94) 25-36, 1994.
- [19] A. Itai and Z. Rosberg. *A golden ratio control policy for a multiple-access channel*. IEEE Trans. Automat. Contr., AC-29, 712-718, 1984.
- [20] C. Kenyon and N. Schabanel. *The data broadcast problem with non-uniform transmission times*. Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 99), 547-556, 1999.
- [21] D. E. Knuth. *The art of computer programming*. Vol. 1, Reading, MA: Addison-Wesley, 1973.
- [22] Z. Rosberg. *Cell Multiplexing in ATM Networks*. IEEE Transactions on Networking, Vol. 4, 112-122, Feb. 1996.
- [23] R. Roundy. *98%-effective integer-ratio lot-sizing for one-warehouse multi-retailer systems*. Management Science, Vol. 31, 1416-1430, 1985.
- [24] C. J. Su and L. Tassiulas. *Broadcast Scheduling for Information Distribution*. Proceedings of the 16th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 97), 109-117, 1997.
- [25] W. Wei and C. Liu. *On a periodic maintenance problem*. Operations Research letters, Vol. 2, 90-93, 1983.
- [26] J. W. Wong. *Broadcast delivery*. in *Proceedings of IEEE* pp. 1566-1577, Dec. 1988.
- [27] N. Vaidya and S. Hameed. *Log time algorithms for scheduling single and multiple channel data broadcast*. Proceedings of the 3rd Annual ACM/IEEE International Conference on Mobile Computing and Networking, 90-99, 1997.
- [28] N. Vaidya and S. Hameed. *Data Broadcast Scheduling: On-line and Off-line Algorithms*. Tech. report, 96-017, Computer Science, Texas A&M univ., July 1996.