Quality of service of data broadcasting algorithms on erroneous wireless channels

Paolo Barsocchi [*] Alan A. Bertossi [†] M. Cristina Pinotti [‡]
Francesco Potortì[*]

Abstract

Broadcasting is an efficient and scalable way of transmitting data over wireless channels to an unlimited number of clients. In this chapter the problem of allocating data to multiple channels is studied, assuming flat data scheduling per channel and the presence of unrecoverable channel transmission errors. The behavior of wireless channels is described by the Bernoulli model, in which each packet transmission has the same probability to fail and each transmission error is independent from the others. The objective is that of minimizing the average expected delay experienced by the clients. Optimal solutions can be found in polynomial time when all data items have unit lengths, while heuristics are presented when data items have non-unit lengths. Extensive simulations, performed on benchmarks whose item popularities follow Zipf distributions, show that good sub-optimal solutions are found.

Keywords: broadcast scheduling algorithm, data allocation algorithm, average expected delay, wireless channel, channel error model, heuristics, quality of service

[*]ISTI-CNR, via G. Moruzzi 1, 56124 Pisa, Italy, {Paolo.Barsocchi,potorti}@isti.cnr.it

[†]Department of Computer Science, University of Bologna, 40127 Bologna, Italy, bertossi@cs.unibo.it

[‡]Department of Computer Science and Mathematics, University of Perugia, 06123 Perugia, Italy, pinotti@unipg.it

1 Introduction

Advances in technology have produced today's information age. Pervasive connections enable a broad spectrum of novel applications and services. In the present environment, voice services are no longer sufficient to satisfy clients' requirements. Access to services on the air seems to be the next killer application. *Data broadcasting* is an efficient way of simultaneously disseminating data items to a large number of clients [15]. In this scenario, a server at the base-station repeatedly transmits data items from a given set over wireless channels, while clients passively listen to the shared channels waiting for their desired item. The server has to pursue a data allocation strategy for assigning items to channels and a broadcast schedule for deciding which item has to be transmitted on each channel at any time instant. The quality of service is measured in terms of the client expected delay, that is, the average amount of time spent by a client before receiving the item he needs. Therefore, efficient data allocation and broadcast scheduling algorithms have to minimize the client expected delay. Such a delay increases with the size of the set of the data items to be transmitted by the server. Indeed, the client has to wait for many unwanted data before receiving his own data. Moreover, the client expected delay may be influenced by transmission errors because items are not always received correctly by the client. Although data are usually encoded using *error correcting codes (ECC)* allowing some recoverable errors to be corrected by the client without affecting the average expected delay, there are several transmission errors which still cannot be corrected using ECC. Such *unrecoverable* errors affect the client expected delay, because the resulting corrupted items have to be discarded and the client must wait until the same item is broadcast again by the server.

Several variants for the problem of data allocation and broadcast scheduling have been proposed in the literature [1, 2, 3, 4, 5, 6, 9, 10, 11, 12, 13, 14, 16, 17, 18].

The database community usually partitions the data among the channels and then adopts a *flat* broadcast schedule on each channel [5, 13, 18]. In such a way, the allocation of data to channels becomes critical for reducing the average expected delay, while the flat schedule on each channel merely consists in cyclically broadcasting in an arbitrary fixed order, that is once at a time in a round-robin fashion, the items assigned to the same channel [1]. In order to reduce the average expected delay, a *skewed* data allocation is used where items are partitioned according to their popularities so that the most requested items appear in a channel with shorter period. Assuming that each item transmitted by the server is always received correctly by the client, a solution that minimizes the average expected delay can be found in polynomial time in the case of *unit lengths* [18], that is when all the items have a unit transmission time, whereas the problem becomes computationally intractable for non-unit lengths [5]. In this latter case, several heuristics have been developed in [4, 18], which have been tested on some benchmarks where item popularities follow Zipf distributions. Such distributions are used to characterize the popularity of one item among a set of similar data, like a web page in a web site [8].

The data allocation problem has not been investigated by the database community when the wireless channels are subject to transmission errors. In contrast, a wireless environment subject to errors has been considered by the networking community, which however concentrates only on finding broadcast scheduling for a single channel to minimize the average expected delay [6, 10, 11, 16]. Indeed, the networking community assumes all items replicated over all channels, and therefore no data allocation to the channels is needed. Although it is still unknown whether a broadcast schedule on a single channel with minimum average delay can be found in polynomial time or not, almost all the proposed solutions follow the *Square Root Rule (SRR)*, a heuristic which in practice finds near-optimal schedules [3]. The aim of Square Root Rule is to produce a broadcast schedule where each data item appears with equally spaced replicas, whose frequency is proportional to the square root of its popularity and inversely proportional to the square root of its length. In particular, the solution proposed by [16] adapts the Square Root Rule solution to the case of unrecoverable errors. In such a case, since corrupted items must be discarded worsening the average expected delay, the spacing among replicas has to be properly recomputed.

The present chapter considers the data allocation problem under the assumptions of skewed data allocation to channels and flat data schedule per channel [4, 5, 18], as studied by the database community, but also copes with the presence of unrecoverable erroneous transmissions, as studied in [7, 16]. The behavior of wireless channels is described by the Bernoulli model, in which each packet transmission has the same probability q to fail and $1 - q$ to succeed, and each transmission error is independent from the others. Specifically, in the case of Bernoulli channel error model, it is shown that an optimum solution, namely one minimizing the average expected delay, can be found in polynomial time for the data allocation problem when the data items have unit lengths. Instead, sub-optimal solutions found by heuristic algorithms are exhibited non-unit lengths. Extensive simulations show that such heuristics provide a good quality of service when tested on benchmarks whose items popularities are characterized by Zipf distributions.

The rest of this chapter is so organized. Section 2 first gives notations, definitions as well as the problem statement, and then reviews the basic algorithms known so far in the case of error-free channel transmissions. Sections 3 considers the Bernoulli channel error model and illustrates how the previously reviewed algorithms can be adapted to cope with erroneous transmissions. Experimental evaluations of the algorithms are reported at the end Sections 3. Finally, conclusions are offered in Section 4.

2 Error-free channels

Consider a set of K identical error-free channels, and a set $D = \{d_1, d_2, \ldots, d_N\}$ of N data items. Each item d_i is characterized by a *popularity* p_i and a *length* z_i, with $1 \leq i \leq N$. The popularity p_i represents the demand probability of item d_i, namely its probability to be requested by the clients, and it does not vary along the time. Clearly, $\sum_{i=1}^{N} p_i = 1$. The length z_i is an integer number, counting how many packets are required to transmit item d_i on any channel and it includes the encoding of the item with an error correcting code. For the sake of simplicity, it is assumed that a packet transmission requires one time unit. Each d_i is assumed to be non preemptive, that is, its transmission cannot be interrupted. When all data lengths are equal to one, i.e., $z_i = 1$ for $1 \leq i \leq N$, the lengths are called *unit* lengths, otherwise they are said to be *non-unit* lengths. The sum of all the item lengths and the maximum item length are denoted, respectively, by Z and z, namely $Z = \sum_{i=1}^{N} z_i$ and $z = max_{1 \leq i \leq N} z_i$.

The *expected delay* t_i is the expected number of packets a client must wait for receiving item d_i. The *average expected delay* (AED) is the number of packets a client must wait on the average for receiving any item, and is computed as the sum over all items of their expected delay multiplied by their popularity, that is

$$\text{AED} = \sum_{i=1}^{N} t_i p_i \tag{1}$$

When the items are partitioned into K groups $G_1, \ldots, G_K$, where group G_k collects the data items assigned to channel k, and a flat schedule is adopted for each channel, that is, the items in G_k are cyclically broadcast in an arbitrary fixed order, Equation 1 can be simplified. Indeed, if item d_i is assigned to channel k, and assuming that clients can start to listen at any instant of time with the same probability, then t_i becomes $\frac{Z_k}{2}$, where Z_k is the schedule *period* on channel k, i.e., $Z_k = \sum_{d_i \in G_k} z_i$. Then, Equation 1 can be rewritten as

$$\text{AED} = \sum_{i=1}^{N} t_i p_i = \sum_{k=1}^{K} \sum_{d_i \in G_k} \frac{Z_k}{2} p_i = \sum_{k=1}^{K} \left(\frac{Z_k}{2} \sum_{d_i \in G_k} p_i \right) = \frac{1}{2} \sum_{k=1}^{K} Z_k P_k \tag{2}$$

where P_k denotes the sum of the popularities of the items assigned to channel k, i.e., $P_k = \sum_{d_i \in G_k} p_i$. Note that, in the unit length case, the period Z_k coincides with the cardinality of G_k, which will be denoted by N_k.

Summarizing, given K error-free channels, a set D of N items, where each data item d_i comes along with its popularity p_i and its integer length z_i, the *Data Allocation Problem* consists in partitioning D into K groups $G_1, \ldots, G_K$, so as to minimize the average expected delay objective function given in Equation 2. Note that, in the special case of unit lengths, the corresponding objective function is derived replacing Z_k with N_k in Equation 2.

Almost all the algorithms proposed so far for the data allocation problem on multiple error-free channels are based on dynamic programming. Such algorithms restrict the search for the solutions to the so called *segmentations*, that is, partitions obtained by considering the items ordered by their indices, and by assigning items with consecutive indices to each channel. Formally, a segmentation is a partition of the ordered sequence $d_1, \ldots, d_N$ into K adjacent segments $G_1, \ldots, G_K$, each of consecutive items, as follows:

$$\underbrace{d_1, \ldots, d_{B_1}}_{G_1}, \underbrace{d_{B_1+1}, \ldots, d_{B_2}}_{G_2}, \ldots, \underbrace{d_{B_{K-1}+1}, \ldots, d_N}_{G_K}$$

A segmentation can be compactly denoted by the $(K-1)$-tuple

$$(B_1, B_2, \ldots, B_{K-1})$$

of its *right borders*, where border B_k is the index of the last item that belongs to group G_k. Notice that it is not necessary to specify B_K, the index of the last item of the last group, because its value will be N for any segmentation.

Almost all the dynamic programming algorithms for multiple channels assume that the items $d_1, d_2, \ldots, d_N$ are indexed by non-increasing $\frac{p_i}{z_i}$ ratios, that is $\frac{p_1}{z_1} \geq \frac{p_2}{z_2} \geq \cdots \geq \frac{p_N}{z_N}$. Observe that for unit lengths this means that the items are sorted by non-increasing popularities. Let $SOL_{k,n}$ denote a segmentation for grouping items $d_1, \ldots d_n$ into k groups and let $sol_{k,n}$ be its corresponding cost, for any $k \leq K$ and $n \leq N$. Moreover, let $C_{i,j}$ denote the cost of assigning to a single channel the consecutive items $d_i, \ldots, d_j$:

$$C_{i,j} = \sum_{h=i}^{j} t_h p_h = \sum_{h=i}^{j} \left(\frac{1}{2} \sum_{h=i}^{j} z_h \right) p_h = \frac{1}{2} \left(\sum_{h=i}^{j} z_h \right) \left(\sum_{h=i}^{j} p_h \right) \tag{3}$$

For unit lengths, the above formula simplifies as $C_{i,j} = \frac{1}{2}(j - i + 1) \sum_{h=i}^{j} p_h$. Note that, once the items are sorted, all the $C_{i,j}$'s can be found in $O(N)$ time by means of prefix-sum computations [17].

The four main algorithms for solving the problem are now briefly surveyed. The first three of them, called *Dynamic-Programming*, *Dichotomic*, and *Dlinear*, assume items sorted by non-increasing $\frac{p_i}{z_i}$'s (and thus they search for segmentations) and work for an arbitrary number of channels. Whereas, the other one, called *Square Root Rule*, do not assume sorted items and works for a single channel. The first three algorithms are off-line and employ dynamic programming, while the last algorithm is on-line and does not use dynamic programming.

2.1 The Dynamic-Programming algorithm

The Dynamic-Programming algorithm is a dynamic programming implementation of the following recurrence, where k varies from 1 to K and, for each fixed k, n varies from 1 to N:

$$sol_{k,n} = \begin{cases} C_{1,n} & \text{if } k = 1 \\ \min_{1 \leq \ell \leq n-1} \{ sol_{k-1,\ell} + C_{\ell+1,n} \} & \text{if } k > 1 \end{cases} \tag{4}$$

For any value of k and n, the Dynamic-Programming algorithm selects the best segmentation obtained by considering the $n-1$ segmentations already computed for the first $k-1$ channels and for the first ℓ items, and by combining each of them with the cost of assigning the last $n-\ell$ items to the single k-th channel. In details, consider the $K \times N$ matrix M with $M_{k,n} = sol_{k,n}$. The entries of M are computed row by row applying Recurrence 4. Clearly, $M_{K,N}$ contains the cost of a solution for the original problem. In order to actually construct the corresponding segmentation, a second matrix F is employed to keep track of the final borders of segmentations corresponding to entries of M. In Recurrence 4, the value of ℓ which minimizes the right-hand-side is the *final border* for the solution $SOL_{k,n}$ and is stored in $F_{k,n}$. Hence, the segmentation is given by $SOL_{K,N} = (B_1, B_2, \ldots, B_{K-1})$ where, starting from $B_K = N$, the value of B_k is equal to $F_{k+1,B_{k+1}}$, for $k = K-1, \ldots, 1$. The Dynamic-Programming algorithm requires $O(N^2K)$ time. It finds an optimal solution in the case of unit lengths and a sub-optimal one in the case of non-unit lengths [18].

2.2 The Dichotomic algorithm

To improve on the time complexity of the Dynamic-Programming algorithm, the Dichotomic algorithm has been devised. Let B_h^n denote the h-th border of $SOL_{k,n}$, with $k > h \geq 1$. Assume that $SOL_{k-1,n}$ has been found for every $1 \leq n \leq N$. If $SOL_{k,l}$ and $SOL_{k,r}$ have been found for some $1 \leq l \leq r \leq N$, then one knows that B_{k-1}^c is between B_{k-1}^l and B_{k-1}^r, for any $l \leq c \leq r$. Thus, choosing c as the middle point between l and r, Recurrence 4 can be rewritten as:

$$sol_{k,\lceil\frac{l+r}{2}\rceil} = \min_{B_{k-1}^l \leq \ell \leq B_{k-1}^r} \{sol_{k-1,\ell} + C_{\ell+1,\lceil\frac{l+r}{2}\rceil}\} \tag{5}$$

where B_{k-1}^l and B_{k-1}^r are, respectively, the final borders of $SOL_{k,l}$ and $SOL_{k,r}$.

Such a recurrence is iteratively solved within three nested loops which vary, respectively, in the ranges $1 \leq k \leq K$, $1 \leq t \leq \lceil \log N \rceil$, and $1 \leq i \leq 2^{t-1}$, and where the indices l, r, and c are set as follows: $l = \lceil \frac{i-1}{2^{t-1}}(N+1) \rceil$, $r = \lceil \frac{i}{2^{t-1}}(N+1) \rceil$, and $c = \lceil \frac{l+r}{2} \rceil = \lceil \frac{2i-1}{2^t}(N+1) \rceil$. In details, the Dichotomic algorithm is shown in Figure 1. It uses the two matrices M and F, whose entries are again filled up row by row (Loop 1). A generic row k is filled in stages (Loop 2). Each stage corresponds to a particular value of the variable t (Loop 3). The variable c corresponds to the index of the entry which is currently being filled in stage t. The variables l (left) and r (right) correspond to the indices of the entries nearest to c which have been already filled, with $l < c < r$. If no entry before c has been already filled, then

$l = 1$, and therefore the final border $F_{k,1}$ is initialized to 1. If no entry after c has been filled, then $r = N$, and thus the final border $F_{k,N+1}$ is initialized to N. To compute the entry c, the variable ℓ takes all values between $F_{k,l}$ and $F_{k,r}$. The index ℓ which minimizes the recurrence in Loop 4 is assigned to $F_{k,c}$, while the corresponding minimum value is assigned to $M_{k,c}$.

The Dichotomic algorithm lowers the time complexity of the Dynamic-Programming algorithm to $O(NK \log N)$. As for the Dynamic-Programming algorithm, the Dichotomic algorithm also finds optimal and sub-optimal solutions for unit and non-unit lengths, respectively [5].

2.3 The Dlinear algorithm

Fixed k and n, the Dlinear algorithm selects the feasible segmentations that satisfy the following Recurrence:

$$sol_{k,n} = \begin{cases} C_{1,n} & \text{if } k = 1 \\ sol_{k-1,m} + C_{m+1,n} & \text{if } k > 1 \end{cases} \tag{6}$$

where

$$m = \min_{B_k^{n-1} \le \ell \le n-1} \left\{ \ell \ : sol_{k-1,\ell} + C_{\ell+1,n} < sol_{k-1,\ell+1} + C_{\ell+2,n} \right\}.$$

In practice, Dlinear adapts Recurrence 4 by exploiting the property that, if $SOL_{k,n-1}$ is known, then one knows that B_k^n is no smaller than B_k^{n-1}, and by stopping the trials as soon as the cost $sol_{k-1,\ell} + C_{\ell+1,n}$ of the solution starts to increase.

The Dlinear algorithm is shown in Figure 2. As before, matrices M and F are used, which are filled row by row. Note that in Loop 1 the leftmost $k - 1$ entries in row k of both M and F are meaningless, since at least one item has to be assigned to each channel. The value of m in Recurrence 6 that gives $M_{k,n}$ is computed iteratively in Loop 3 and stored in $F_{k,n}$.

The overall time complexity of the Dlinear algorithm is $O(N(K + \log N))$. Thus the Dlinear algorithm is even faster than the Dichotomic one, but the solutions it provides are always sub-optimal, both in the unit and non-unit length case [4].

2.4 The Square Root Rule algorithm

When there is only one channel, the Dynamic-Programming, Dichotomic, and Dlinear algorithms provide a trivial flat schedule with period Z. In such a case, each t_i is equal to $\frac{Z}{2}$ and hence also the average expected delay is equal to $\frac{Z}{2}$, regardless of the item popularities. To overcome this drawback, a schedule is needed where the spacing between two consecutive transmissions of one item is not the same for all items, but depends on both the popularity and the length of such an item.

It has been shown in [16] that, in an optimal schedule, replicas of any item d_i should be equally spaced with spacing

$$s_i = \left(\sum_{h=1}^{N} \sqrt{p_h z_h} \right) \sqrt{\frac{z_i}{p_i}} \tag{7}$$

In this way, the expected delay for item d_i becomes half of its spacing and thus, substituting $t_i = \frac{s_i}{2}$ in Equation 1, the average expected delay becomes

$$\text{AED} = \frac{1}{2} \left(\sum_{i=1}^{N} \sqrt{p_i z_i} \right)^2 \tag{8}$$

The average expected delay value given in Equation 8 represents a lower bound which in general is not achievable because the replicas cannot always be kept equally spaced. The Square Root Rule algorithm is an on-line heuristic which tries to keep the replicas as equally spaced as possible. For this purpose, it determines the item to be transmitted next by using the decision rule $\frac{s_i^2 p_i}{z_i} = \text{constant}$, based on Equation 7. Let T denote the current time, let R_i be the time at which the last replica of d_i has been transmitted (initialized to -1), and let $G_i = (T - R_i)^2 \frac{p_i}{z_i}$, where $T - R_i$ is the spacing for item d_i if d_i would be transmitted again at time T. At each instant of time T, the Square Root Rule algorithm evaluates the decision rule G_i for all items d_i, $1 \leq i \leq N$, selects for transmission at time T that item d_h with maximum G_h, and finally updates $R_h = T$ and $T = T + z_h$.

The Square Root Rule algorithm takes $O(N)$ time to select the item to be transmitted. Such a time can be reduced to $O(M)$ by partitioning the items into M buckets according to their G's values [16].

3 Bernoulli channel error model

In this section, unrecoverable channel transmission errors modeled by a geometric distribution are taken into account. Under such an error model, each packet transmission over every channel has the same probability q to fail and $1-q$ to succeed, and each transmission error is independent from the others, with $0 \leq q \leq 1$. Since the environment is asymmetric, a client cannot ask the server to immediately retransmit an item d_i which has been received on channel k with an unrecoverable error. Indeed, the client has to discard the item and then has to wait for a whole period Z_k, until the next transmission of d_i scheduled by the server. Even the next item transmission could be corrupted, and in such a case an additional delay of Z_k has to be waited. Therefore, the expected delay t_i has to take into account the extra waiting time due to a possible sequence of independent unrecoverable errors.

3.1 Unit length items

Assume that the items have unit lengths, i.e., $z_i = 1$ for $1 \leq i \leq N$. Recall that in such a case the period of channel k is N_k. If a client wants to receive item d_i, which is transmitted on channel k, and the first transmission he can hear of d_i is error-free, then the client waits on the average $\frac{N_k}{2}$ time units with probability $1-q$. Instead, if the first transmission of d_i is erroneous, but the second one is error-free, then the client experiences an average delay of $\frac{N_k}{2} + N_k$ time units with probability $q(1-q)$. Generalizing, if there are h bad transmissions of d_i followed by a good one, the client average delay for receiving item d_i becomes $\frac{N_k}{2} + hN_k$ time units with probability $q^h(1-q)$. Thus, summing up over all h, the expected delay t_i is equal to

$$\sum_{h=0}^{\infty} (\frac{N_k}{2} + hN_k) q^h (1-q) = \frac{N_k}{2} + N_k \frac{q}{1-q}$$

because $\sum_{h=0}^{\infty} q^h = \frac{1}{1-q}$ and $\sum_{h=0}^{\infty} hq^h = \frac{q}{(1-q)^2}$. Therefore, one can set the expected delay as

$$t_i = \frac{N_k}{2} \frac{1+q}{1-q} \tag{9}$$

By the above setting, the objective function to be minimized becomes

$$\text{AED} = \sum_{i=1}^{N} t_i p_i = \frac{1}{2} \frac{1+q}{1-q} \sum_{k=1}^{K} N_k P_k \tag{10}$$

Therefore, for items with unit lengths, the data allocation problem can be optimally solved in polynomial time. This derives from Lemmas 1 and 2 of [5] which prove optimality in the particular case of error-free channels, that is, when $q = 0$. Indeed, when $q > 0$, similar proofs hold once the cost $C_{i,j}$ of assigning consecutive items $d_i, \ldots, d_j$ to the same channel is defined as $C_{i,j} = \frac{j-i+1}{2} \frac{1+q}{1-q} \sum_{h=i}^{j} p_h$. In words, Lemmas 1 and 2 of [5] show that, whenever the items $d_1, d_2, \ldots, d_N$ are sorted by non-increasing popularities, there always exists an optimal solution which is a segmentation and which can be found by the Dichotomic algorithm.

3.2 Non-unit length items

Consider now items with non-unit lengths and recall that Z_k is the period of channel k. In order to receive an item d_i of length z_i over channel k, a client has to listen for z_i consecutive error-free packet transmissions, which happens with probability $(1-q)^{z_i}$. Hence, the failure probability for item d_i on channel k is $Q_{z_i} = 1 - (1-q)^{z_i}$.

In the case that the first transmission of d_i heard by the client is error-free, the client has to wait on the average $\frac{Z_k}{2}$ time units with probability $1 - Q_{z_i}$. Instead, the client waits on the average for $\frac{Z_k}{2} + Z_k$ time units with probability $Q_{z_i}(1 - Q_{z_i})$ in the case that the first transmission of d_i is erroneous and the second one is error-free. In general, h bad transmissions of d_i followed by a good one lead to a delay of $\frac{Z_k}{2} + hZ_k$ time units with probability $Q_{z_i}^h(1 - Q_{z_i})$. Therefore, summing up over all h as seen in the unit length case, the expected delay becomes

$$t_i = \frac{Z_k}{2} \frac{1 + Q_{z_i}}{1 - Q_{z_i}} \tag{11}$$

Thus, the average expected delay to be minimized is

$$\text{AED} = \frac{1}{2} \sum_{k=1}^{K} \left(Z_k \sum_{d_i \in G_k} \frac{1 + Q_{z_i}}{1 - Q_{z_i}} p_i \right) \tag{12}$$

Recalling that the items are indexed by non-increasing $\frac{p_i}{z_i}$ ratios, the new recurrences for the Dichotomic and Dlinear algorithms are derived from Recurrences 5 and 6, respectively, once each $C_{i,j}$ is defined as $C_{i,j} = \frac{1}{2} \left(\sum_{h=i}^{j} z_h \right) \left(\sum_{h=i}^{j} \frac{1+Q_{z_h}}{1-Q_{z_h}} p_h \right)$. All the $C_{i,j}$'s can be computed in $O(N)$ time via prefix-sums, once $O(H)$ time is spent for computing all the Q_{z_h}'s, where $H = \min\{N \log z, z\}$. Therefore, the time complexities of the Dichotomic and Dlinear algorithms become, respectively, $O(NK \log N + H)$ and $O(N(K + \log N) + H)$. Note that in such a case optimality is not guaranteed since the problem is computationally intractable already for error-free channels.

When there is only one channel, it has been shown in [16] that, in an optimal schedule, replicas of any item d_i should be equally spaced with spacing

$$s_i = \left(\sum_{h=1}^{N} \sqrt{p_h z_h \frac{1 + Q_{z_h}}{1 - Q_{z_h}}} \right) \sqrt{\frac{z_i}{p_i} \frac{1 - Q_{z_i}}{1 + Q_{z_i}}} \tag{13}$$

Thus, substituting $t_i = \frac{s_i}{2}$ in Equation 1, the average expected delay becomes

$$\text{AED} = \frac{1}{2} \left(\sum_{i=1}^{N} \sqrt{p_i z_i \frac{1 + Q_{z_i}}{1 - Q_{z_i}}} \right)^2 \tag{14}$$

Therefore, the Square Root Rule algorithm can be applied once the decision rule G_i is modified as $G_i = (T - R_i)^2 \frac{p_i}{z_i} \frac{1+Q_{z_i}}{1-Q_{z_i}}$.

3.3 Quality of service evaluation

In this subsection, the behavior of the Dichotomic, Dlinear, and Square Root Rule heuristics is evaluated in the case of Bernoulli channel error model. The above algorithms have been experimentally tested on benchmarks where the item popularities follow a Zipf distribution. Specifically, given the number N of items and a real number $0 \leq \theta \leq 1$, the item popularities are defined as

$$p_i = \frac{(1/i)^\theta}{\sum_{h=1}^{N} (1/h)^\theta} \qquad 1 \leq i \leq N$$

In the above formula, θ is the *skew* parameter. In particular, $\theta = 0$ stands for a uniform distribution with $p_i = \frac{1}{N}$, while a higher θ implies a higher skew, namely the difference among the p_i values becomes larger.

Consider first some experiments for multiple channels reported from [7], where either the skew parameter θ is set to 0.8 as suggested in [18], $N = 2500$, and $10 \leq K \leq 500$, or $\theta = 0.8$, $K = 50$, and $500 \leq N \leq 2500$, or $0 \leq \theta \leq 1$, $N = 2500$, and $K = 200$. The item lengths z_i are integers randomly generated according to a uniform distribution in the range $1 \leq z_i \leq 10$, for $1 \leq i \leq N$. The channel failure probabilities can assume the values 0.001 and 0.01.

Moreover, since the data allocation problem is computationally intractable when items have non-unit lengths, lower bounds for a non-unit length instance are derived by transforming it into a unit length instance as follows. Each item d_i of popularity p_i and length z_i is decomposed into z_i items of popularity $\frac{p_i}{z_i}$ and length 1. Since more freedom has been introduced, it is clear that the optimal average expected delay for the so transformed problem is a lower bound on the average expected delay of the original problem. Since the transformed problem has unit lengths, the optimal average expected delay can be obtained by running the polynomial time Dichotomic algorithm both when all the channels are error-free or have the same failure probability.

Figures 3-5 show the experimental results for the Dichotomic and Dlinear algorithms in the case that there are multiple channels, the items have non-unit lengths, and the failure probability q is 0.001. One can note that the two above mentioned lower bounds as well as the solutions provided by both algorithms almost coincide. Instead, Figures 6-8 show the experimental results when the failure probability q is 0.01. Referring to Figures 6 and 7, where $\theta = 0.8$, the average expected delay of the transformed unit length instance in the presence of errors is $\frac{1+q}{1-q} = 1.02$ times the average expected delay of the same transformed instance without errors. One can also note that, since the average item length is 5, the average expected delay of the original instance in the presence of errors should be about $\frac{1+Q}{1-Q} = 1.10$ times the average expected delay of the same original instance in the absence of errors, where $Q = 1 - (1 - 0.01)^5 = 0.05$. This can be easily checked in Figure 6, e.g., for $K = 10$, where the ratio between the two values of the average expected delay is about $\frac{500}{450} = 1.11$. Referring to Figure 8, where θ varies, one notes that the ratio between such average expected delay values is almost 1.12 for every value of θ, confirming the results of Figures 6 and 7.

Consider now some simulation experiments for a single channel, which are reported from [16]. In the experiments, $N = 1000$, $0 \leq \theta \leq 1$, and each z_i is an integer randomly generated according to a uniform distribution in the range $1 \leq z_i \leq 10$, for $1 \leq i \leq N$. The channel failure probability q varies between 0 and 0.2. Figure 9 shows the behavior of the Square Root Rule algorithm compared with the analytical lower bound given in Equation 14. The experimental tests show that the average expected delay values obtained by the Square Root Rule algorithm and by the lower bound differ up to 3% for small values of q, and up to 10% for larger values of q.

4 Conclusions

This chapter considered the problem of allocating data to multiple channels, assuming skewed allocation of most popular data items to less loaded channels, flat data scheduling per channel, and the presence of unrecoverable channel transmission errors. The quality of service was measured in terms of the average expected delay experienced by the clients. The behavior of some heuristics has been experimentally evaluated when modelling the channel error by means of the Bernoulli model. Extensive simulations showed that such heuristics give sub-optimal solutions which provide a good quality of service, when tested on benchmarks whose item popularities follow Zipf distributions. In particular, for small channel error probabilities, the average expected delay of the proposed solutions is almost the same as the optimal one found in the case of channels without errors.

Acknowledgement

This work has been supported by ISTI-CNR under the BREW research grant. The C++ code used in the simulations was written by G. Spagnardi.

References

[1] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik. Broadcast disks: data management for asymmetric communication environments. In *Proc. SIGMOD*, May 1995.

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

[3] M.H. Ammar and J.W. Wong. On the optimality of cyclic transmission in teletext systems. *IEEE Transactions on Communications*, 35(11):1159–1170, 1987.

[4] S. Anticaglia, F. Barsi, A.A. Bertossi, L. Iamele, and M.C. Pinotti. Efficient heuristics for data broadcasting on multiple channels. *Wireless Networks*, 2006, published on line.

[5] E. Ardizzoni, A.A. Bertossi, M.C. Pinotti, S. Ramaprasad, R. Rizzi, and M.V.S. Shashanka. Optimal skewed data allocation on multiple channels with flat broadcast per channel. *IEEE Transactions on Computers*, 54(5):558–572, 2005.

[6] A. Bar-Noy, R. Bhatia, J.S. Naor, and B. Schieber. Minimizing service and operation costs of periodic scheduling. In *Proc. Ninth ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 11–20, 1998.

[7] P. Barsocchi, A.A. Bertossi, M.C. Pinotti, and F. Potortí. Data broadcasting over error-prone wireless channels. *Technical Report* 2006/9, Department of Mathematics and Computer Science, University of Perugia, Italy, 2006.

[8] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and Zipf-like distributions: evidence and implications. In *Proc. IEEE INFOCOM*, 1999.

[9] T. Imielinski, S. Viswanathan, and B.R. Badrinath. Energy efficient indexing on air. In *Proc. SIGMOD*, May 1994.

[10] C. Kenyon and N. Schabanel. The data broadcast problem with non-uniform transmission time. In *Proc. Tenth ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 547–556, 1999.

[11] C. Kenyon, N. Schabanel, and N. Young. Polynomial time approximation scheme for data broadcast. In *Proc. ACM Symp. on Theory of Computing (STOC)*, pages 659–666, 2000.

[12] S.-C. Lo and A.L.P. Chen Optimal index and data allocation in multiple broadcast channels. In *Proc. Sixteenth IEEE Int'l Conf. on Data Engineering (ICDE)*, February 2000.

[13] W.C. Peng and M.S. Chen. Efficient channel allocation tree generation for data broadcasting in a mobile computing environment. *Wireless Networks*, 9(2):117–129, 2003.

[14] K.A. Prabhakara, K.A. Hua, and J. Oh. Multi-level multi-channel air cache designs for broadcasting in a mobile environment. In *Proc. Sixteenth IEEE Int'l Conf. on Data Engineering (ICDE)*, February 2000.

[15] I. Stojmenovic (Editor). *Handbook of Wireless Networks and Mobile Computing.* Wiley, Chichester, 2002.

[16] N. Vaidya and S. Hameed. Log time algorithms for scheduling single and multiple channel data broadcast. In *Proc. Third ACM-IEEE Conf. on Mobile Computing and Networking (MOBICOM)*, September 1997.

[17] W.G. Yee, Efficient data allocation for broadcast disk arrays. *Technical Report* GIT-CC-02-20, Georgia Institute of Technology, 2001.

[18] W.G. Yee, S. Navathe, E. Omiecinski, and C. Jermaine. Efficient data allocation over multiple channels at broadcast servers. *IEEE Transactions on Computers*, 51(10):1231–1236, 2002.

```
Input:        $N$ items sorted by non-increasing $\frac{p_i}{z_i}$ ratios, and $K$ groups;
Initialize:       for $i$ from 1 to $N$ do
                      for $k$ from 1 to $K$ do
                          if $k = 1$ then $M_{k,i} \leftarrow C_{k,i}$ else $M_{k,i} \leftarrow \infty$;
Loop 1:           for $k$ from 2 to $K$ do
                      $F_{k,0} \leftarrow F_{k,1} \leftarrow 1$; $F_{k,N+1} \leftarrow N$;
Loop 2:               for $t$ from 1 to $\lceil \log N \rceil$ do
Loop 3:                   for $i$ from 1 to $2^{t-1}$ do
                              $c \leftarrow \lceil \frac{2i-1}{2^t}(N+1) \rceil$;  $l \leftarrow \lceil \frac{i-1}{2^{t-1}}(N+1) \rceil$;  $r \leftarrow \lceil \frac{i}{2^{t-1}}(N+1) \rceil$;
                              if $M_{k,c} = \infty$ then
Loop 4:                           for $\ell$ from $F_{k,l}$ to $F_{k,r}$ do
                                      if $M_{k-1,\ell} + C_{\ell+1,c} < M_{k,c}$ then
                                          $M_{k,c} \leftarrow M_{k-1,\ell} + C_{\ell+1,c}$;
                                          $F_{k,c} \leftarrow \ell$;
```

Figure 1: *The Dichotomic algorithm.*

```
Input:        $N$ items sorted by non-increasing $\frac{p_i}{z_i}$ ratios, and $K$ groups;
Initialize:   for $n$ from 1 to $N$ do
                  $M_{1,n} \leftarrow C_{1,n}$;
Loop 1:       for $k$ from 2 to $K$ do
                  $F_{k,k} \leftarrow k-1$;
                  $M_{k,k} \leftarrow M_{k-1,k-1} + C_{k,k}$;
Loop 2:           for $n$ from $k+1$ to $N$ do
                      $\ell \leftarrow F_{k,n-1}$;
                      $m \leftarrow \ell$;
                      $M_{k,n} \leftarrow M_{k-1,\ell} + C_{\ell+1,n}$;
                      $incr \leftarrow$ false;
Loop 3:               while $\ell \leq n-2$ and $\neg$ $incr$ do
                          $temp \leftarrow M_{k-1,\ell+1} + C_{\ell+2,n}$;
                          if $M_{k,n} \geq temp$ then
                              $M_{k,n} \leftarrow temp$;
                              $\ell \leftarrow \ell + 1$;
                          else
                              $incr \leftarrow$ true;
                      $m \leftarrow \ell$;
                      $F_{k,n} \leftarrow m$
```

Figure 2: *The Dlinear algorithm.*

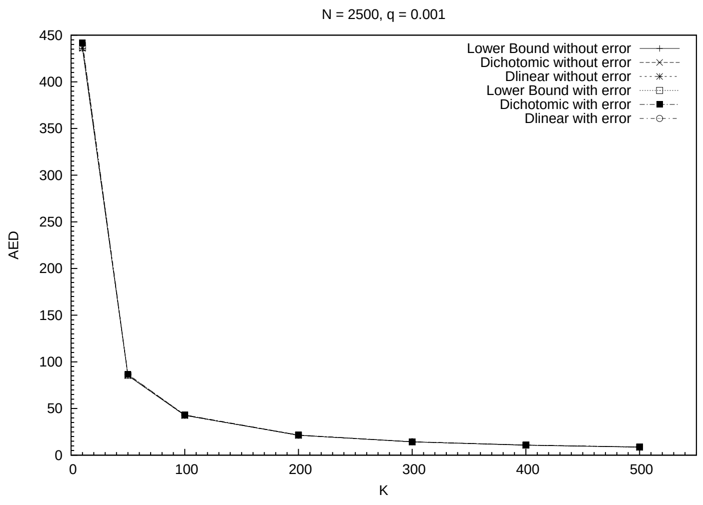

Figure 3: *Results for* 2500 *items of non-unit lengths, when* $\theta = 0.8$ *and the* K *channels have failure probability* $q = 0.001$.

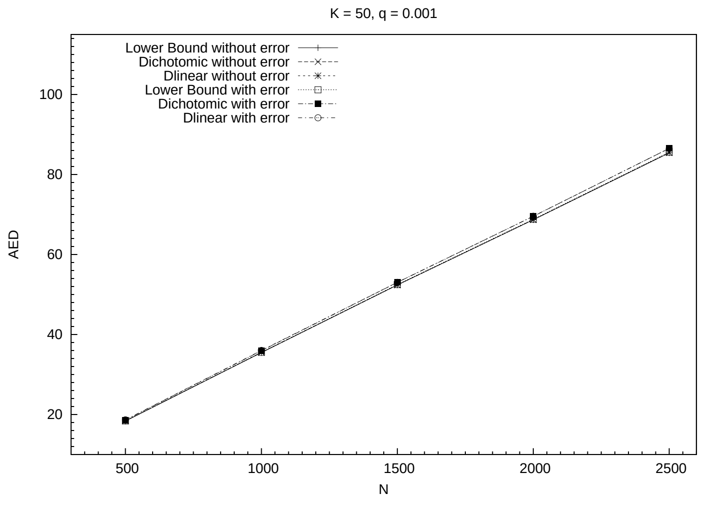

Figure 4: *Results for* N *items of non-unit lengths, when* $\theta = 0.8$ *and the* 50 *channels have failure probability* $q = 0.001$.

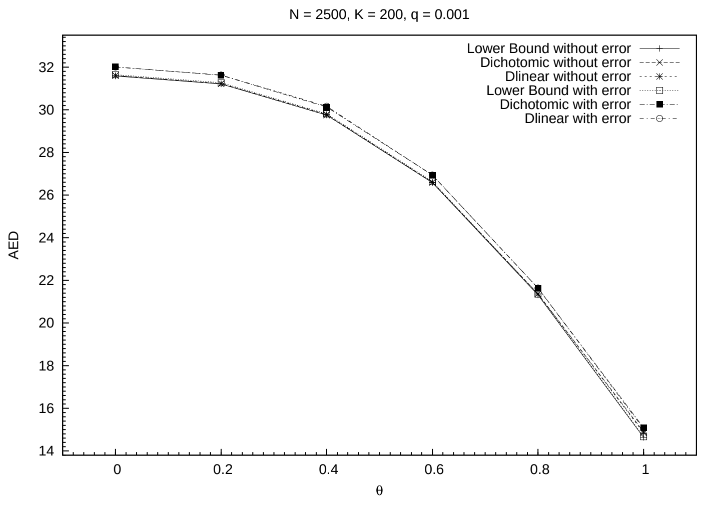

Figure 5: *Results for* 2500 *items of non-unit lengths, when* $0 \leq \theta \leq 1$ *and the* 200 *channels have failure probability* $q = 0.001$.

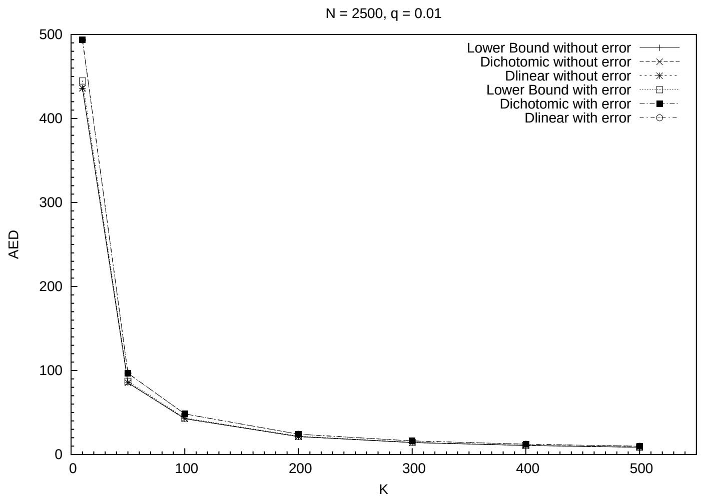

Figure 6: *Results for* 2500 *items of non-unit lengths, when* $\theta = 0.8$ *and the* K *channels have failure probability* $q = 0.01$.

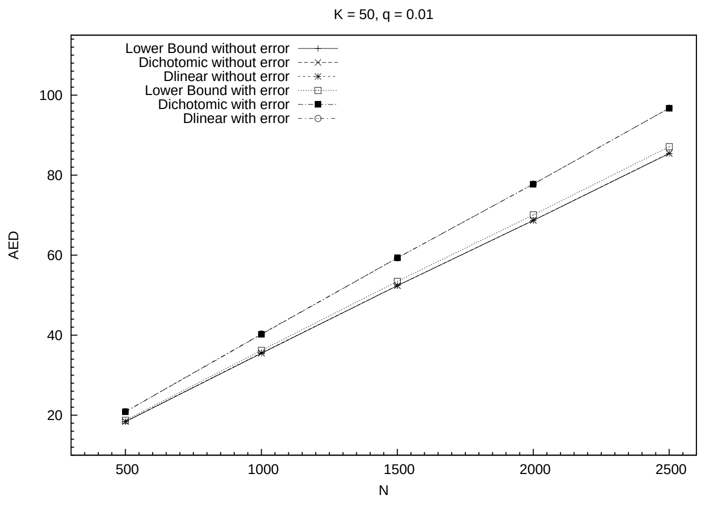

Figure 7: *Results for N items of non-unit lengths, when $\theta = 0.8$ and the 50 channels have failure probability $q = 0.01$.*

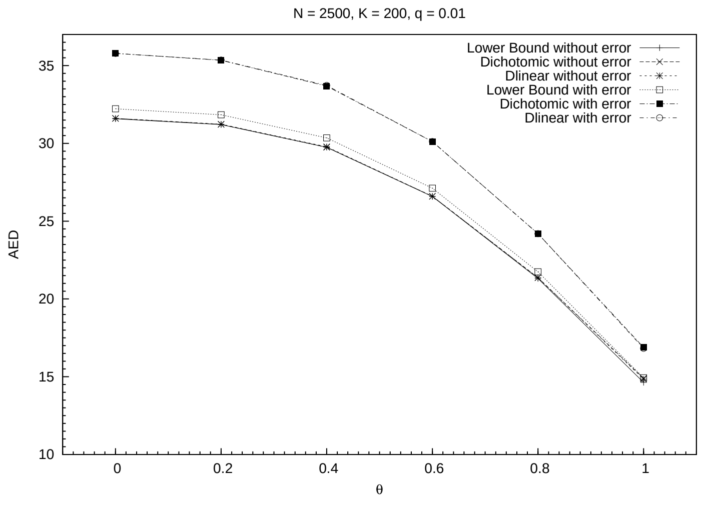

Figure 8: *Results for 2500 items of non-unit lengths, when $0 \leq \theta \leq 1$ and the 200 channels have failure probability $q = 0.01$.*

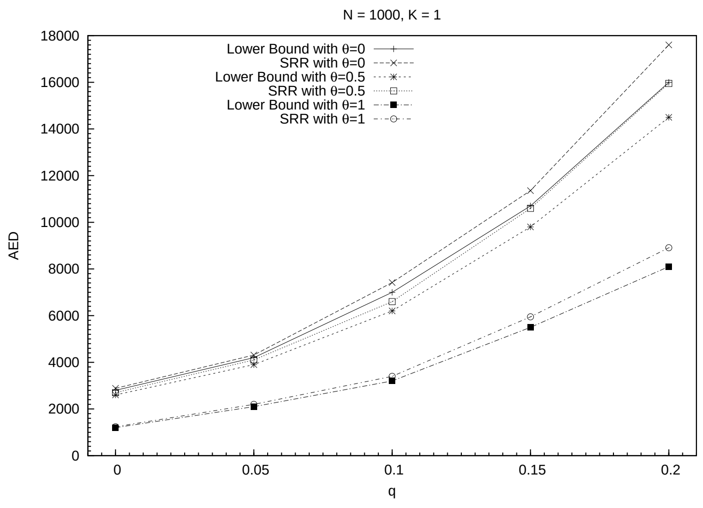

Figure 9: *Results for* 1000 *items with non-unit lengths, when* $K = 1$, $0 \leq \theta \leq 1$, *and* $0 \leq q \leq 0.2$.*(In the legend, SRR stands for Square Root Rule).*