Skewed Allocation of Non-Uniform Data for Broadcasting over Multiple Channels

A.A. Bertossi[1], C.M. Pinotti[2]

[1]University of Bologna
Dept. of Computer Science
Mura Anteo Zamboni 7, 40127 Bologna
bertossi@cs.unibo.it

[2]University of Perugia
Dept. of Math. & Computer Science
Via Vanvitelli 1, 06123 Perugia
pinotti@unipg.it

Abstract

The problem of data broadcasting over multiple channels consists in partitioning data among channels, depending on data popularities, and then cyclically transmitting them over each channel so that the average waiting time of the clients is minimized. Such a problem is known to be polynomially time solvable for uniform length data items, while it is computationally intractable for non-uniform length data items. In this paper, two new heuristics are proposed which exploit a novel characterization of optimal solutions for the special case of two channels and data items of uniform lengths. Sub-optimal solutions for the most general case of an arbitrary number of channels and data items of non-uniform lengths are provided. The first heuristic, called Greedy+*, combines the novel characterization with the known greedy approach, while the second heuristic, called* Dlinear*, combines the same characterization with the dynamic programming technique. Such heuristics have been tested on benchmarks whose popularities are characterized by Zipf distributions. The experimental tests reveal that Dlinear finds optimal solutions almost always, requiring good running times, while Greedy+ is faster and scales well when changes occur on the input parameters, but provides worse solutions than Dlinear.*

Keywords: Wireless communication, data broadcasting, multiple channels, heuristics, flat scheduling, average waiting time, dynamic programming.

1 Introduction

Broadcasting is an efficient way of simultaneously disseminating data to a large number of clients. It is especially effective in an asymmetric wireless environment, where a server at a base-station repeatedly transmits data items from a given set over multiple wireless channels, while clients wait for their desired item on the proper channel [1, 2, 7, 8]. A careful allocation strategy has to be pursued to assign data items to channels so as to reduce the waiting time of the clients. Typically, items are partitioned in a skewed manner among channels according to their popularities so that the most requested items appear in a channel with shorter broadcast period [4, 10]. Moreover, for each single channel, the server follows a schedule to decide which item to broadcast at any time instant. Usually, a *flat* schedule is adopted which fixes an arbitrary order among the data items allocated to the same channel and transmits the items once at a time, in a round-robin fashion.

The problem of data broadcasting over multiple channels, with the objective of minimizing the average waiting time of the clients, under the assumptions of skewed allocation to channels and flat scheduling per channel, has been introduced in [10] and expanded in [4]. The problem has been shown to be polynomial time solvable for uniform length data items [10], and it has been proved to be computationally intractable (NP-hard) for non-uniform length data items [4]. In these papers, several algorithms have been proposed, all of which assume that a sorting preprocessing step has been done on the data items.

In the uniform case, the fastest known algorithm producing an optimal solution requires $O(NK \log N)$ time [4], where N is the number of items and K is

1-4244-0054-6/06/$20.00

the number of channels. Such an optimal algorithm is based on dynamic programming and solves NK problem instances, for $1 \leq n \leq N$ and $1 \leq k \leq K$. In the non-uniform case, the problem can be optimally solved in pseudo-polynomial time when $K = 2$, by a reduction to a knapsack problem, and in exponential time for arbitrary K [4]. In this latter case, a heuristic, called *Greedy*, has been proposed in [10]. Fixed N, Greedy starts with all data items assigned to one channel, and then proceeds by splitting the items of one channel between two channels, thus adding a new channel, until K channels are reached. In practice, Greedy is very fast, scales with the number of channels, and provides fair sub-optimal solutions for the K instances of the problem, where N is fixed and $1 \leq k \leq K$.

This paper presents two new heuristics which provide sub-optimal solutions for the data broadcasting problem with non-uniform data item lengths and an arbitrary number of channels. As for Greedy, both heuristics assume that the items are sorted by decreasing popularities per length unit. As opposed to Greedy, they pretend that a novel characterization of the optimal solution of the problem for $K = 2$ and uniform lengths holds also for the general case of arbitrary K and non-uniform lengths. The first heuristic, called *Greedy+*, follows the same strategy as Greedy. To solve K instances with N fixed and $1 \leq k \leq K$, it requires $O(N(K + \log N))$ time in the worst case. The second heuristic, called *Dlinear*, follows the dynamic programming relation proposed in [10] and requires $O(N(K + \log N))$ time for solving all the NK instances, for $1 \leq n \leq N$ and $1 \leq k \leq K$. The proposed heuristics are experimentally tested on some benchmarks, whose popularities follow a Zipf distribution. Such a distribution has been shown to characterize the popularity of one element among a set of similar data, like a web page in a web site [5]. The experimental tests reveal that the quality of the solution provided by Dlinear is much better than that produced by the other heuristics. Indeed, Dlinear finds optimal solutions almost always, requiring reasonable running times. Although Greedy remains the fastest heuristic, it gives the worst sub-optimal solutions. Both the running times and the quality of the solutions of Greedy+ are intermediate between those of Dlinear and Greedy. Like Greedy, Greedy+ scales well with respect to all parameters changes.

The rest of this paper is so organized. Section 2 gives notations, definitions and the problem statement. In particular, Subsection 2.1 proves the novel characterization of the optimal solution, for the special case with $K = 2$ and uniform lengths, that motivates the new heuristics. Section 3 presents the $O(N(K + \log N))$ time Greedy+ and Dlinear heuristics. Moreover, it is also shown that the Greedy+ algorithm, when restricted to uniform length data, can be implemented so as to take $O((N + K) \log N)$ time in the worst case. Section 4 reports the experimental tests of the heuristics, performed on randomly generated instances. Finally, conclusions are offered in Section 5.

2 Preliminaries

Consider a set of K identical 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 how frequently item d_i is requested by the clients, and it does not vary along the time. Popularities can be either arbitrary positive integers, or real numbers normalized in the range $(0, 1]$ such that $\sum_{i=1}^{N} p_i = 1$. The length z_i is an integer number, counting how many time units (or, ticks) are required to transmit item d_i on any channel. When all data lengths are the same, i.e. $z_i = z$ for $1 \leq i \leq N$, the lengths are called *uniform* and are assumed to be unit, i.e. $z = 1$. When the data lengths are not the same, the lengths are said *non-uniform*.

The items have to be partitioned into K groups $G_1, \ldots, G_K$. Group G_j collects the data items assigned to channel j, with $1 \leq j \leq K$. The cardinality of G_j is denoted by N_j, the sum of its item lengths is denoted by Z_j, i.e. $Z_j = \sum_{d_i \in G_j} z_i$, and the sum of its popularities is denoted by P_j, i.e. $P_j = \sum_{d_i \in G_j} p_i$. Note that since the items in G_j are cyclically broadcast according to a flat schedule, Z_j is the schedule period on channel j. Clearly, in the uniform case $Z_j = N_j$, for $1 \leq j \leq K$. If item d_i is assigned to channel j, and assuming that clients can start to listen at any instant of time with the same probability, the *client expected delay* for receiving item d_i is half of the period, namely $\frac{Z_j}{2}$. Assuming, as in [10], that indexing allows clients to know in advance the content of the channels, the *average expected delay* (AED) over all channels is

$$\text{AED} = \frac{1}{2} \sum_{j=1}^{K} Z_j P_j \tag{1}$$

Given K 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 broadcasting problem consists in partitioning D into K groups $G_1, \ldots, G_K$, so as to minimize the objective function AED given in Equation 1. In the special case of equal lengths, the corresponding objective function is derived replacing Z_j with N_j in Equation 1.

Some known results, proposed in [10, 4], that will be used in the next sections, are now briefly recalled.

Lemma 1. [10] *Let G_h and G_j be two groups in an optimal solution for a problem instance with uniform lengths. Let d_i and d_k be items with $d_i \in G_h$ and $d_k \in G_j$. If $N_h < N_j$, then $p_i \geq p_k$. Similarly, if $p_i > p_k$, then $N_h \leq N_j$.*

In other words, the most popular items are allocated to less loaded channels so that they appear more frequently. As a consequence, if the items are sorted by non-increasing popularities, then the group sizes are non-decreasing.

Corollary 1. [10] *Let $d_1, d_2, \ldots, d_N$ be N uniform length items with $p_i \geq p_k$ whenever $i < k$. Then, there exists an optimal solution for partitioning them into K groups $G_1, \ldots, G_K$, where each group is made of consecutive elements.*

By the above corollary, in the uniform case the items are assumed to be sorted by non-increasing popularities, and any solution S will be compactly represented by a *segmentation*, that is a $(K-1)$-tuple $(B_1, B_2, \ldots, B_{K-1})$, where B_j is the index, called *border*, of the rightmost item belonging to group G_j and $B_1 < B_2 < \ldots < B_{K-1}$. Notice that the cardinality of G_j, i.e. the number N_j of items in the group, is $N_j = B_j - B_{j-1}$, where $B_0 = 0$ and $B_K = N$ are assumed. From now on, a segmentation $S = (B_1, B_2, \ldots, B_{K-1})$ for the uniform case is called *feasible* if $N_1 \leq N_2 \leq \ldots \leq N_K$. Indeed, by Lemma 1, an optimal solution will be sought only among feasible solutions.

For any $n \leq N$ and $k \leq K$, let $opt_{n,k}$ denote the cost of an optimal solution for items $d_1, \ldots d_n$ and k channels (groups). Let $C_{i,h}$ be the cost of assigning consecutive items $d_i, \ldots, d_h$ to one group, i.e. $C_{i,h} = \frac{1}{2}(h-i+1)\sum_{q=i}^{h} p_q$. The following result holds.

Theorem 1. [10] *Let $d_1, d_2, \ldots, d_N$ be N uniform length items, sorted by non-increasing popularities. Hence,*

$$opt_{n,k} = \begin{cases} C_{1,n} & \text{if } k = 1 \\ \min_{1 \leq \ell \leq n-1}\{opt_{\ell,k-1} + C_{\ell+1,n}\} & \text{if } k > 1 \end{cases} \quad (2)$$

Theorem 1 suggests an $O(N^2K)$ time dynamic programming algorithm to solve the problem in the uniform case. Indeed, consider the $K \times N$ matrix M with $M_{k,n} = opt_{n,k}$. The entries of M are computed row by row applying Recurrence 2. In order to actually construct an optimal partition, a second matrix F is employed which stores in $F_{k,n}$ the value of ℓ which minimizes the right-hand-side of Equation 2. Hence, the optimal solution for N and K is given by $S = (B_1, B_2, \ldots, B_{K-1})$ where, starting from $B_K = N$, the value of B_k is equal to $F_{k+1,B_{k+1}}$, for $k = K-1, \ldots, 1$. A useful property of the optimal solution is that the values stored in each row of matrix F are non-decreasing, as stated in the following Lemma:

Lemma 2. [4] *Let $d_1, d_2, \ldots, d_N$ be N uniform length items, sorted by non-increasing popularities. For any $n \leq N$ and $k \leq K$, $F_{k,n-1} \leq F_{k,n}$.*

In words, Lemma 2 implies that, given the items sorted by non-increasing popularities, if one builds an optimal solution for N items from an optimal solution for $N-1$ items, then the border B_{K-1} can only move to the right.

2.1 Two Channels and Uniform Lengths

This subsection exploits the structure of the optimal solution in the special case where the item lengths are uniform and there are only two channels. Indeed, as shown later, the values assumed varying ℓ in the right hand side of Recurrence 2 for $k = 2$ form a *unimodal* sequence. That is, there is a particular index ℓ such that the values on its left are in non-increasing order, while those on its right are in increasing order. By this fact, one can search the minimum of Recurrence 2 in a very effective way, improving on the overall running time.

Formally, for $K = 2$ and N uniform length data items, the problem is to find a partition S into G_1 and G_2 such that $AED_S = \frac{1}{2}(N_1P_1 + N_2P_2)$ is minimized, where N_i and P_i denote the cardinality and the sum of the popularities of items in G_i, respectively, with $i = 1, 2$. Clearly, $N = N_1 + N_2$, and by Lemma 1, $N_1 \leq N_2$ holds for any optimal solution. Moreover, any feasible solution S can be denoted by the single border B_1, which coincides with N_1.

Lemma 3. *Consider N uniform length items, sorted by non-increasing popularities, and $K = 2$ channels. Let $S = (N_1)$ be a feasible solution such that $P_1 \leq P_2$. If the solution $S' = (N_1 + 1)$ is feasible, then $AED_{S'} \leq AED_S$.*

Proof. Since S' is feasible, then $N_1 + 1 \leq N_2 - 1$. The new solution S' differs from S because item d_{N_1+1} is moved from G_2 to G_1. Therefore, $AED_{S'} = \frac{1}{2}((N_1 + 1)(P_1 + p_{N_1+1}) + (N_2 - 1)(P_2 - p_{N_2-1})) = \frac{1}{2}(N_1P_1 + N_2P_2 + (N_1 - N_2 + 2)p_{N_1+1} + (P_1 - P_2))$.

Since $AED_S = \frac{1}{2}(N_1P_1 + N_2P_2)$, $N_1 - N_2 + 2 \leq 0$, and $P_1 - P_2 \leq 0$, it follows that $AED_{S'} \leq AED_S$. □

While Lemma 1 gives the upper bound $N_1 \leq \lfloor \frac{N}{2} \rfloor$ on the cardinality of group G_1, Lemma 3 provides a lower bound b on N_1. Indeed, it guarantees that any optimal solution contains at least the first b items $d_1, \ldots, d_b$, where b is the largest index for which $P_1 = \sum_{h=1}^{b} p_h \leq P_2 = \sum_{h=b+1}^{N} p_h$. Formally, Recurrence 2 for $K = 2$ can be rewritten as follows:

$$opt_{N,2} = \min_{b \leq \ell \leq \lfloor \frac{N}{2} \rfloor} \{C_{1,\ell} + C_{\ell+1,N}\} \quad (3)$$

where

$$b = \max_{1 \leq s \leq \lfloor \frac{N}{2} \rfloor} \left\{ s \ : \ \sum_{h=1}^{s} p_h \leq \sum_{h=s+1}^{N} p_h \right\}.$$

The following lemma improves on the upper bound of N_1 given by Lemma 1, and shows that the values of the feasible solutions assumed in the right-hand side of Equation 3 form a unimodal sequence.

Lemma 4. *Consider N uniform length items, sorted by non-increasing popularities, and $K = 2$ channels. Let $S = (N_1)$ be a feasible solution such that $P_1 > P_2$. Consider the solutions $S' = (N_1{+}1)$ and $S'' = (N_1{+}2)$. If $AED_{S'} > AED_S$, then $AED_{S''} > AED_{S'}$.*

Proof. By definition, $AED_S = \frac{1}{2}(N_1 P_1 + N_2 P_2)$ and $AED_{S'} = \frac{1}{2}((N_1+1)(P_1 + p_{N_1+1}) + (N_2 - 1)(P_2 - p_{N_1+1})) = AED_S + \frac{1}{2}((P_1 - P_2) + p_{N_1+1}(N_1 - N_2 + 2))$.

Since $AED_{S'} > AED_S$, it follows that $(P_1 - P_2) > p_{N_1+1}(N_2 - N_1 - 2)$.

Moreover, $AED_{S''} = \frac{1}{2}(N_1 + 2)(P_1 + p_{N_1+1} + p_{N_1+2}) + \frac{1}{2}(N_2 - 2)(P_2 - p_{N_1+1} - p_{N_1+2})$, and thus $AED_{S''} - AED_{S'} = \frac{1}{2}(P_1 - P_2) + \frac{1}{2}p_{N_1+2}(N_1 - N_2 + 2) + (p_{N_1+1} + p_{N_1+2})$.

Since $p_{N_1+1} \geq p_{N_1+2}$, one has $(P_1 - P_2) > p_{N_1+2}(N_2 - N_1 - 2)$.

Finally, $AED_{S''} > AED_{S'}$ holds because $\frac{1}{2}(P_1 - P_2) + \frac{1}{2}p_{N_1+2}(N_1 - N_2 + 2) + (p_{N_1+1} + p_{N_1+2}) > \frac{1}{2}(P_1 - P_2) + \frac{1}{2}p_{N_1+2}(N_1 - N_2 + 2) > 0$. □

In practice, one can scan the feasible solutions of Equation 3 by moving the border ℓ rightwards, one position at a time, starting from the lower bound b obtained applying Lemma 3. The scan continues while the average expected delay of the current solution does not increase, but stops as soon as the average expected delay starts to increase. Indeed, by Lemma 4, further moving the border ℓ to the right can only increase the cost of the solutions. Hence the border m that minimizes Equation 3, that is the optimal solution of the problem, is given by:

$$opt_{N,2} = C_{1,m} + C_{m+1,N} \quad (4)$$

```
Procedure BinSearch (i, j);
    m ← ⌊(i+j)/2⌋;
    if i = j then
        return m
    else
        if f(m) ≥ f(m+1) then
            BinSearch (m+1, j)
        else
            BinSearch (i, m);
```

Figure 1. The binary search on a unimodal sequence.

where

$$m = \min_{b \leq \ell \leq \lfloor \frac{N}{2} \rfloor} \{\ell \ : C_{1,\ell} + C_{\ell+2,N} < C_{1,\ell+1} + C_{\ell+2,N}\}.$$

Note that, in the above equation, the cost variation is:

$$(C_{1,\ell+1} + C_{\ell+2,N}) - (C_{1,\ell} + C_{\ell+1,N}) =$$

$$\frac{1}{2}(\sum_{h=1}^{\ell} p_h - \sum_{h=\ell+1}^{N} p_h + p_{\ell+1}(2\ell + 2 - N)).$$

Due to the unimodal property of the sequence of values on the right-hand side of Equation 4, the search of m can be done in $O(\log N)$ time by a suitable modified binary search [6]. For the sake of completeness, a simplified implementation of such binary search is given in the following.

Let $f(\ell) = C_{1,\ell} + C_{\ell+1,N} = \frac{\ell}{2}\sum_{h=1}^{\ell} p_h + \frac{N-\ell}{2}\sum_{h=\ell+1}^{N} p_h$. Then, the unimodal sequence consists of the values $f(b), f(b+1), \ldots, f(\lfloor \frac{N}{2} \rfloor)$. As said, solving Equation 4 is equivalent to find the index m such that $f(b) \geq \ldots \geq f(m) < f(m+1) < \ldots < f(\lfloor \frac{N}{2} \rfloor)$. This can be done by invoking the recursive procedure *BinSearch*, given in Figure 1, with parameters $i = b$ and $j = \lfloor \frac{N}{2} \rfloor$. The BinSearch procedure first computes the middle point $m = \lfloor \frac{i+j}{2} \rfloor$. Then, the values $f(m)$ and $f(m+1)$ are compared in the light of the unimodal sequence definition. If $f(m) \geq f(m+1)$, the minimum must belong to the right half, otherwise it must be in the left half. Procedure BinSearch proceeds recursively on the proper half until the minimum is reached.

3 New Heuristics

The purpose of the new heuristics to be presented in this section is to quickly find good sub-optimal solutions for the most general case of non-uniform data

lengths and an arbitrary number of channels. Such a goal is achieved by pretending that the optimal solution characterization, proved in Subsection 2.1 for the special case of two channels and uniform lengths, holds also in the general case of more than two channels and non-uniform lengths.

As for all the previously known heuristics, the new heuristics also assume that the items are sorted by non-increasing $\frac{p_i}{z_i}$ ratios. This can be done in $O(N \log N)$ time by a sorting preprocessing step. Moreover, since the lengths are non-uniform, the cost of assigning the items from d_i to d_j to a single channel becomes $C_{i,j} = \frac{1}{2}\left(\sum_{h=i}^{j} p_h\right)\left(\sum_{h=i}^{j} z_h\right)$. Letting $P_{i,j} = \sum_{h=i}^{j} p_h$ and $Z_{i,j} = \sum_{h=i}^{j} z_h$, one notes that all the $P_{1,n}$ and $Z_{1,n}$, for $1 \leq n \leq N$, can be computed in $O(N)$ time by two prefix sum computations, performed as a preprocessing step. Hence, a single $C_{i,j}$ can be computed on the fly in constant time as $C_{i,j} = \frac{1}{2}(P_{1,j} - P_{1,i-1})(Z_{1,j} - Z_{1,i-1})$. From now on, in order to simplify the presentation, $C_{i,j}$ is defined to be 0 whenever $i > j$.

3.1 The Greedy+ Algorithm

The Greedy+ heuristic is a refinement of the Greedy heuristic presented in [9]. Recall that the Greedy heuristic works for a fixed number N of data items. It initially assigns all the N items to a single group. Then, for $K-1$ times, one of the groups is split in two groups, that will be assigned to two different channels. To find which group to split along with its actual split point, all the possible points of all groups are considered as split point candidates, and the one that decreases AED the most is selected. An efficient implementation takes advantage from the fact that, between two subsequent splits, it is sufficient to recompute the costs for the split point candidates of the last group that has been actually split.

In summary, Greedy+ consists of two phases. In the first phase it behaves as Greedy, except for the way the split point is determined. In the second phase, the solution provided by the first phase is refined by working on pairs of consecutive channels.

Specifically, in the first phase, Greedy+ uses an approach similar to that of Equation 4 to determine the split point. This is because splitting one channel is the same as solving the data broadcast problem for two channels. In details, assume that the channel to be split contains the items from d_i to d_j, with $1 \leq i < j \leq N$, and let $cost_{i,j,2}$ denote the cost of a feasible solution for assigning such items to two channels. Then, the split point is given by the value of m that satisfies the following relation:

$$cost_{i,j,2} = C_{i,m} + C_{m+1,j} \tag{5}$$

where

$$m = \min_{i \leq \ell \leq j-1} \{\ell \ : C_{i,\ell} + C_{\ell+1,j} < C_{i,\ell+1} + C_{\ell+2,j}\}.$$

Note that, since the item lengths are not uniform, the sequence of values $C_{i,\ell} + C_{\ell+1,j}$, for $i \leq \ell \leq j-1$, is not unimodal. However, Greedy+ behaves as such a sequence were unimodal. Hence, instead of trying all the possible values of ℓ between i and j, as done by Greedy, Greedy+ performs a left-to-right scan starting from i and stopping as soon the AED increases. In this way, a sub-optimal solution $S = (B_1, B_2, \ldots . B_{K-1})$ is found.

The second phase is performed only when $K \geq 3$ and consists in refining the solution S by recomputing its borders. It consists in a sequence of odd steps, followed by a sequence of even steps. During the t-th odd step, $1 \leq t \leq \lfloor \frac{K}{2} \rfloor$, the two-channel subproblem including the items assigned to groups G_{2t-1} and G_{2t} is solved. Specifically, Equation 5 is applied choosing $i = B_{2t-2} + 1$ and $j = B_{2t}$, thus recomputing the border B_{2t-1} of S. Similarly, during the t-th even step, $1 \leq t \leq \lfloor \frac{K-1}{2} \rfloor$, the two-channel subproblem including the items assigned to groups G_{2t} and G_{2t+1} is solved by applying Equation 5 with $i = B_{2t-1}+1$ and $j = B_{2t+1}$, thus recomputing the border B_{2t} of S.

The initial sorting requires $O(N \log N)$ time. Since each split runs in $O(N)$ time, and K splits are computed, the first phase of Greedy+ takes $O(NK)$ time. The second phase of Greedy+ requires $O(N)$ time since each item is considered as a candidate split point at most in a single split computation among all the odd steps, and in a single split computation among the even steps. Therefore, the overall time required in the worst case by the Greedy+ heuristic is $O(N(K + \log N))$, the same as the original Greedy heuristic proposed in [10].

In the special case of uniform data lengths, by Lemma 4, each split is performed on a unimodal sequence by invoking the BinSearch procedure, which takes $O(\log N)$ time. Therefore, the worst case time complexity of Greedy+ becomes $O((N+K) \log N)$, improving over the $O(N(K + \log N))$ time of the original Greedy algorithm [10].

Note that Greedy+ scales well when changes occur on the number of channels, on the number of items, on item popularities, as well as on item lengths. Indeed, adding or removing a channel simply requires doing a new split or removing the last introduced split, respectively. Adding a new item first requires to insert such an item in the sorted item sequence. Assume the new item is added to group G_j, then the border

of the two-channel subproblem including items of G_j and G_{j+1} is recomputed by applying Equation 5. Similarly, deleting an item that belongs to group G_j requires to solve again the two-channel subproblem including items of G_j and G_{j+1}. Finally, a change in the popularity/length of an item is equivalent to first removing that item and then adding the same properly modified item.

3.2 The Dlinear Algorithm

The Dlinear heuristic follows a dynamic programming approach similar to that provided by Recurrence 2. It solves all the NK instances, for $1 \leq n \leq N$ and $1 \leq k \leq K$, with the objective of obtaining an $O(N(K+\log N))$ worst case time complexity. Fixed k, Dlinear computes a solution for n items from the previously computed solution for $n-1$ items, exploiting the characteristics of the optimal solutions for the uniform case.

For any $n \leq N$ and $k \leq K$, let $M_{k,n}$ denote the cost of a feasible solution for items $d_1, \ldots d_n$ and k channels, and let $F_{k,n}$ be the index of the last element assigned to channel $k-1$ in such a solution. Dlinear selects the feasible solutions that satisfy the following Recurrence:

$$M_{k,n} = \begin{cases} C_{1,n} & \text{if } k = 1 \\ M_{k-1,m} + C_{m+1,n} & \text{if } k > 1 \end{cases} \qquad (6)$$

where

$$m = \min_{F_{k,n-1} \leq \ell \leq n-1} \{\ell \ : M_{k-1,\ell} + C_{\ell+1,n} < M_{k-1,\ell+1} + C_{\ell+2,n}\}.$$

In practice, Dlinear pretends to adapt Recurrence 4, that holds for the uniform data lengths, also to the case of non-uniform data lengths. In particular, the choice of the lower bound $F_{k,n-1}$ in the formula of m is suggested by Lemma 2 which says that the border of channel $k-1$ can only move right when a new item with the smallest popularity is added. Moreover, m is determined as in Equation 4 pretending that the sequence $M_{k-1,\ell} + C_{\ell+1,n}$, obtained for $F_{k,n-1} \leq \ell \leq n-1$, be unimodal. Therefore, the solution provided by Dlinear is a sub-optimal one.

As regard to the time complexity, computing $M_{k,n}$ requires $O(F_{k,n} - F_{k,n-1})$ time. Hence, row k of M is filled in $\sum_{n=1}^{N} O(F_{k,n} - F_{k,n-1}) = O(F_{k,N} - F_{k,1}) = O(N)$ time. Since M has K rows and the sorting step takes $O(N \log N)$ time, the overall time complexity of the Dlinear algorithm is $O(N(K + \log N))$.

$N/K/\theta/z$	**Algorithm**	**AED**	**% Error**	**Time**
500/20/0.8/3	Greedy	18.72	7.1	102
	Greedy+	17.58	0.6	3514
	Dlinear	17.47		2106
	Lower bound	17.47		
1500/20/0.8/3	Greedy	53.85	7.9	283
	Greedy+	51.71	3.6	21240
	Dlinear	49.90		6519
	Lower bound	49.90		
1750/20/0.8/3	Greedy	62.64	7.9	326
	Greedy+	58.92	1.5	31137
	Dlinear	58.04		7488
	Lower bound	58.04		
2000/20/0.8/3	Greedy	71.24	7.9	373
	Greedy+	66.93	1.4	38570
	Dlinear	65.98		8602
	Lower bound	65.98		
2250/20/0.8/3	Greedy	79.70	7.8	457
	Greedy+	75.06	1.6	45170
	Dlinear	73.87		9749
	Lower bound	73.87		
2500/20/0.8/3	Greedy	88.40	7.8	474
	Greedy+	82.51	0.7	62376
	Dlinear	81.93		10920
	Lower bound	81.93		

Table 1. Experimental results on Zipf distributions, when $K = 20$, $\theta = 0.8$, and $z = 3$.

4 Experimental Tests

In this section, experimental results, performed on implementations of both the Greedy+ and Dlinear heuristics, are discussed for the data broadcasting problem with K channels and non-uniform lengths. In addition, the implementation of Greedy, as detailed in [9], is used for comparison purposes. The algorithms are written in C and the experiments are run on an AMD Athlon XP 2500+, 1.84 GHz, with 1 GB RAM.

The heuristics are tested on some non-uniform length instances generated as follows. Given the number N of items and a real number $0 \leq \theta \leq 1$, the item popularities are generated according to a Zipf distribution whose skew is θ, namely:

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

In the above formula, $\theta = 0$ stands for a uniform distribution with $p_i = \frac{1}{N}$, while $\theta = 1$ implies a high skew, namely the range of p_i values becomes larger. The item lengths z_i are integers generated according to a uniform distribution in the range $1 \leq z_i \leq z$, as in [8]. The items are sorted by non-increasing $\frac{p_i}{z_i}$ ratios, as suggested in [8]. The parameters N, K, z, and θ vary, respectively, in the ranges: $500 \leq N \leq 2500$, $10 \leq K \leq 500$, $3 \leq z \leq 10$, and $0.5 \leq \theta \leq 1$.

$N/K/\theta/z$	Algorithm	AED	% Error	Time
2500/10/0.8/3	Greedy	179.16	7.8	381
	Greedy+	167.86	1.0	97356
	Dlinear	166.14		4919
	Lower bound	166.14		
2500/40/0.8/3	Greedy	44.04	7.9	562
	Greedy+	41.58	1.9	34147
	Dlinear	40.79		22771
	Lower bound	40.79		
2500/80/0.8/3	Greedy	21.98	7.9	685
	Greedy+	20.72	1.7	19179
	Dlinear	20.37		46545
	Lower bound	20.37		
2500/100/0.8/3	Greedy	17.14	5.2	740
	Greedy+	16.75	2.8	27452
	Dlinear	16.29		57906
	Lower bound	16.29		
2500/200/0.8/3	Greedy	8.56	5.1	1009
	Greedy+	8.37	2.8	12974
	Dlinear	8.15	0.1	116265
	Lower bound	8.14		
2500/500/0.8/3	Greedy	3.4	4.2	2313
	Greedy+	3.35	2.7	21430
	Dlinear	3.32	1.8	273048
	Lower bound	3.26		

Table 2. Experimental results on a Zipf distribution, when $N = 2500$, $\theta = 0.8$, and $z = 3$.

Since the optimal solutions can be found in a reasonable time only for small values of N and z, a *lower bound* on AED is used for large values of N and z. The lower bound for a non-uniform instance is obtained by transforming it into a uniform 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 AED for the so transformed problem is a lower bound on the AED of the original problem. Since the transformed problem has uniform lengths, its optimal AED is obtained by running the Dichotomic algorithm presented in [4].

The simulation results are exhibited in Tables 1, 2, 3, and 4. The tables report the time (measured in microseconds), the AED, and the percentage of error, which is computed as

$$\left(\frac{\mathrm{AED}_{\mathrm{heuristic}} - \mathrm{AED}_{\mathrm{lowerbound}}}{\mathrm{AED}_{\mathrm{lowerbound}}}\right) 100$$

The running times reported in the tables do not include the time for sorting.

By observing the tables, one notes that Greedy+ and Dlinear always outperform Greedy in terms of solution quality. In particular, Greedy+ at least halves the error of Greedy, producing solutions whose errors is at most 5.7%. Moreover, Dlinear reaches the optimum almost in all cases, and its maximum error is as high as 1.8% only in one instance.

$N/K/\theta/z$	Algorithm	AED	% Error	Time
2500/50/0.5/3	Greedy	47.74	9.7	595
	Greedy+	46.02	5.7	23175
	Dlinear	43.52	0.02	29075
	Lower bound	43.51		
2500/50/0.7/3	Greedy	39.59	6.8	600
	Greedy+	38.47	3.8	23606
	Dlinear	37.05	0.02	29132
	Lower bound	37.04		
2500/50/0.8/3	Greedy	34.33	5.2	603
	Greedy+	33.49	2.6	24227
	Dlinear	32.61		29121
	Lower bound	32.61		
2500/50/1/3	Greedy	23.10	3.2	609
	Greedy+	22.53	0.6	27566
	Dlinear	22.38		28693
	Lower bound	22.38		

Table 3. Experimental results on Zipf distributions, when $N = 2500$, $K = 50$, and $z = 3$.

As regard to the running times, although all the three heuristics have the same $O(N(K + \log N))$ time, Greedy is the fastest in practice. Greedy+ and Dlinear are slower than Greedy, but their running times are always less than one tenth of second. Their highest running times occur in Table 2, where those of Dlinear are directly proportional to K while those of Greedy+ are inversely proportional to K. This singular behaviour of Greedy+ might depend on the fact that, in the second phase each Split execution stops its scan earlier when the cardinality of each pair of channels decreases, and therefore the number of channels K increases. It is worth to note that, due to the dynamic programming approach, Dlinear solves all the instances with $1 \leq n \leq N$ items and $1 \leq k \leq K$ channels, while Greedy and Greedy+ only solve the K instances with $n = N$.

Further simulation results for an extensive set of experiments have been reported in [3], which include distributions other than Zipf and/or instances with uniform data lengths. Overall Dlinear still continues to produce the best solutions among all the three heuristics.

5 Conclusions

In this paper, the problem of broadcasting data with non-uniform lengths over multiple channels, with the objective of minimizing the average expected delay of the clients, was considered under the assumptions of skewed allocation to multiple channels and flat schedul-

$N/K/\theta/z$	**Algorithm**	**AED**	**% Error**	**Time**
500/50/0.8/3	Greedy	7.34	5.3	147
	Greedy+	7.19	3.1	2517
	Dlinear	6.98	0.1	5423
	Lower bound	6.97		
500/50/0.8/5	Greedy	10.78	5.3	147
	Greedy+	10.52	2.8	2938
	Dlinear	10.25	0.1	5490
	Lower bound	10.23		
500/50/0.8/7	Greedy	14.50	4.9	146
	Greedy+	14.16	2.4	3329
	Dlinear	13.85	0.2	5499
	Lower bound	13.82		
500/50/0.8/10	Greedy	19.48	5.1	145
	Greedy+	18.97	2.3	3899
	Dlinear	18.58	0.2	5507
	Lower bound	18.53		

Table 4. Experimental results on a Zipf distribution, when $N = 500$, $K = 50$, and $\theta = 0.8$.

ing per channel. Since for non-uniform lengths the problem is computationally intractable, new heuristics have been proposed, which experimentally outperform the previously known heuristic in terms of the solution quality. In particular, the experimental tests have shown that the Dlinear heuristic finds optimal solutions almost always. In contrast, Greedy is the fastest heuristic, but produces the worst solutions. Finally, Greedy+ presents running times and sub-optimal solutions which are both intermediate between those of Greedy and Dlinear. In conclusion, the choice among the heuristics depends on the goal to be pursued. If one is interested in finding the best sub-optimal solutions, then Dlinear should be adopted. Instead, if the running time is the main concern, then Greedy should be chosen, while if adaptability to parameter changes is the priority, then either Greedy or Greedy+ should be applied. In this scenario, Greedy+ represents a good compromise since it is scalable and produces fairly good solutions.

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. On the optimality of cyclic transmission in teletext systems. *IEEE Transactions on Communications*, 35(11):1159–1170, 1987.

[3] S. Anticaglia, F. Barsi, A.A. Bertossi, L. Iamele, and M.C. Pinotti. Efficient Heuristics for Data Broadcasting on Multiple Channels. *Technical Report*, 2005/5, Department of Mathematics and Computer Science, University of Perugia, 2005.

[4] 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.

[5] 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.

[6] A.S. Goldstein and E.M. Reingold. A Fibonacci version of Kraft's inequality applied to discrete unimodal search. *SIAM Journal on Computing*, 22(4):751–777, 1993.

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

[8] 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.

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

[10] 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.