

Maximum Bandwidth Broadcast in Single and Multi-Interface Networks

A.A. Bertossi
Dept. of Computer Science
University of Bologna
40127 Bologna, Italy
bertossi@cs.unibo.it

A. Navarra, C.M. Pinotti
Dept. of Comp. Sci. & Math.
University of Perugia
06123 Perugia, Italy
{navarra,pinotti}@dmi.unipg.it

ABSTRACT

A maximum bandwidth broadcast problem in multi-interface networks is considered, where each transmission, simultaneously serving a group of nodes, achieves a total bandwidth equal to the product between the smallest node bandwidth in the group and the cardinality of the group. Two variants of the problem are studied, called the *K-Maximum Bandwidth Broadcast in Single-Interface Networks (MBB)* and *K-Maximum Bandwidth Broadcast in Multiple-Interface Networks (MBBM)* problems. It is shown that the former problem can be optimally solved in polynomial time, while the latter one is computationally intractable (i.e. NP-hard). Polynomial time algorithms are devised for optimally solving some particular cases of *MBBM* where a common order holds and the number of used interfaces is polylogarithmic in the number of nodes.

Categories and Subject Descriptors

C.2 [Computer-communication Netw.]; C.2.2 [Network Protocols]: [Broadcast]; F.2 [Analysis of Algorithms and Problem Complexity]

General Terms

Algorithms

Keywords

Multi-Interface Networks, Wireless Networks, Broadcast, Maximum Bandwidth, Dynamic Programming, NP-completeness

1. INTRODUCTION

Broadcasting is certainly one of the most important operations when dealing with communication networks. Such an operation is usually provided among the basic primitives implemented in a network in order to set up the normal behavior of a network or to make some upgrades. According to the considered model and environment, many algorithms have been proposed so far to solve the broadcast problem. Here we are interested in multi-interface networks where a

source node s can communicate with all the nodes of the network by means of different interfaces. Each node admits a maximum receiving bandwidth for each interface it holds. In particular, for each interface i and for each node j , the value $b_{i,j}$ represents the maximum bandwidth at which node j can receive by means of interface i . Note that, by setting $b_{i,j} = 0$ we may assume that node j does not hold interface i . The broadcast operation allows s to transmit concurrently over all possible interfaces, but each node can receive only from one interface at a time. Moreover, for each interface used by s , the maximum bandwidth of the corresponding transmission must be the smallest one among all the values associated to the nodes receiving via such an interface. The goal is then to decide which nodes receive over which interface in order to maximize the transmission bandwidth of s over all the interfaces chosen for the transmission task. This choice might depend on various parameters related to the employed technology, available interfaces, or input variables. The problem is certainly of practical interest but its formalization also represents an intriguing combinatorial problem.

At the best of our knowledge, only a few papers have considered a broadcasting model of multi-interface networks in which each transmission, simultaneously serving a group of nodes, has an overall bandwidth equal to the product between the smallest node bandwidth in the group and the cardinality of the group. Indeed, such a model has been used in the local rate maximization problem, one of the four stages of the distributed and localized heuristics presented in [5, 6] for computing distributed 2-hop trees for broadcasting in multi-radio multi-rate multi-channel wireless mesh networks.

In this paper, we formally introduce two variants of the maximum bandwidth broadcast problem, called the *K-Maximum Bandwidth Broadcast in Single-Interface Networks (MBB)* and *K-Maximum Bandwidth Broadcast in Multiple-Interface Networks (MBBM)* problems. The former problem is defined in Section 2, where polynomial time algorithms are also proposed to optimally solve it. The latter problem is formulated in Section 3, where it is shown that it is computationally intractable (i.e. NP-hard) in general and polynomial time algorithms are devised for optimally solving some particular cases. Specifically, *MBBM* is polynomially time solvable when there are at most two interfaces, while it can be reduced to a *Resource Constrained Shortest Path (RCSP)* problem when a *common* order holds (*CMBBM*). Although

RCSP is also NP-hard in general, *CMBBM* is polynomial when the number of used interfaces is polylogarithmic in the number of nodes.

2. SINGLE INTERFACE

Let S be a set of N nodes, each equipped with a unique interface to communicate with the other nodes. For each node j , let b_j be the maximum bandwidth that can be used to communicate toward node j . A transmission that simultaneously serves a subset of nodes $A_i \subseteq S$ is assumed to transfer data at a rate equal to $\min_{j \in A_i} \{b_j\}$. Given a number K of communications, with $K \leq N$, the *K-Maximum Bandwidth Broadcast in Single-Interface Networks* (*MBB* for short) can be stated as follows.

MBB: *K-Maximum Bandwidth Broadcast in Single-Interface Networks*

Input: A set S of N nodes, a bandwidth function $b : S \rightarrow \mathbb{R}_0^+$ and an integer $K \leq N$.

Solution: A partition A_1, A_2, \dots, A_K of S which associates each subset of nodes to one different communication.

Goal: Maximize $\sum_{i=1}^K |A_i| \min_{j \in A_i} \{b_j\}$.

In words, we want to determine which subsets of nodes the source s (external to the network) has to reach simultaneously in order to maximize the bandwidth achievable by a broadcast transmission which performs at most K communications.

In the following, let $\min_i = \min_{j \in A_i} \{b_j\}$ denote the maximum bandwidth achievable when the group A_i of nodes is reached with the same communication and let the set of nodes $S = \{1, 2, \dots, N\}$ be indexed in such a way that $b_i \geq b_j$ whenever $i < j$.

LEMMA 1. *Let $S = \{1, 2, \dots, N\}$ be the set of nodes indexed in such a way that $b_i \geq b_j$ whenever $i < j$. Then, there exists an optimal solution for the MBB problem which partitions the nodes into K groups A_1, \dots, A_K , where each group is made of consecutive nodes.*

PROOF. Consider an optimal solution σ whose groups are not made of consecutive nodes. Order the partition A_1, \dots, A_K of σ in such a way that $\min_i \geq \min_j$, for $1 \leq i < j \leq K$. Clearly, $\min_K = \min_{1 \leq j \leq N} \{b_j\} = b_N$, i.e. $N \in A_K$. Let A_t be the group not made of consecutive nodes with the largest index. Let ℓ and p be, respectively, the smallest node which is missing in the group A_t and the smallest node in A_t out of order. Note that the node with bandwidth \min_t cannot be out of order, or equivalently it has index $N - \sum_{j=t+1}^K |A_j|$, because A_{t+1}, \dots, A_K consist of consecutive nodes and $\min_1 \geq \dots \geq \min_K$. Thus, the missing node has index $\ell \leq N - \sum_{j=t+1}^K |A_j| - 1$. Clearly $b_p \geq b_\ell \geq b_{N - \sum_{j=t+1}^K |A_j|} = \min_t$. Denoted with A_{t_ℓ} the group to which ℓ belongs in σ , and recalling that ℓ is the smallest node out of order, $\min_{t_\ell} = b_\ell$ holds. Now, exchange node ℓ with node p . The new solution σ' achieves a transmission bandwidth larger than or equal to that of σ

because the minimum bandwidth in group A_t remains the same, while \min_{t_ℓ} cannot decrease. Repeating the above process until no node out of order can be found, we build an optimal solution whose groups are made of consecutive nodes. \square

Hereafter, thus, it is assumed that the nodes are sorted by non-increasing bandwidth, and the optimal solutions will be sought within the class of *segmentations*. A *segmentation* is a partition A_1, \dots, A_K , such that if $i \in A_t$ and $p \in A_t$ then $h \in A_t$ whenever $i \leq h \leq p$. A segmentation

$$\underbrace{1, \dots, B_1}_{A_1}, \underbrace{B_1 + 1, \dots, B_2}_{A_2}, \dots, \underbrace{B_{K-1} + 1, \dots, N}_{A_K}$$

will be more compactly denoted by the $(K - 1)$ -tuple

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

of its *right borders*, where border B_j is the index of the last node that belongs to group A_j . Notice that it is not necessary to specify B_K , the index of the last node of the last group, because its value will be N for any solution. From now on, B_{K-1} will be referred to as the *final border* of the solution. The cardinality of B_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. Clearly, in each group $b_{B_j} = \min_j$.

2.1 Polynomial time algorithms

For any two integers $n \leq N$ and $k \leq K$, let $OPT_{n,k}$ denote an optimal solution for grouping nodes $1, \dots, n$ into k groups and let $opt_{n,k}$ be its corresponding cost. Let $C_{i,h}$ be the cost of assigning consecutive nodes i, \dots, h to one group, i.e. $C_{i,h} = (h - i + 1) \min_{i \leq t \leq h} b_t = (h - i + 1)b_h$. Hence, $opt_{n,1} = C_{1,n} = nb_n$ for every n . For $k > 1$, the following recurrence holds:

$$opt_{n,k} = \max_{\ell \in \{1, 2, \dots, n-1\}} \{opt_{\ell, k-1} + C_{\ell+1, n}\} \quad (1)$$

By the recurrence in Equation 1, a dynamic programming algorithm, say *DP*, can be immediately derived to solve the *MBB* problem.

Indeed, in order to find $OPT_{N,K}$, consider the $K \times N$ matrix M with $M_{K,N} = opt_{N,K}$. The entries of M are computed row by row by applying the recurrence of Equation 1. Clearly, $M_{K,N}$ contains the cost of an optimal solution for the *MBB* problem. In order to actually construct an optimal partition, a second matrix F is employed to keep track of the final borders of segmentations corresponding to entries of M . In the recurrence of Equation 1, the value of ℓ which maximizes the right-hand-side is the *final border* for the solution $OPT_{n,k}$ and is stored in $F_{k,n}$. Hence, the optimal segmentation with K communications is given by $OPT_{N,K} = (B_1, B_2, \dots, B_{K-1})$ where, starting from $B_K = N$, the value of B_k is equal to F_{k+1, B_k} , for $k = 1, \dots, K - 1$.

LEMMA 2. *Let $S = \{1, 2, \dots, n\}$ be the set of nodes indexed in such a way that $b_i \geq b_j$ whenever $i < j$. Then, the optimal solution for the MBB problem which partitions the nodes into k groups A_1, \dots, A_k has cost no smaller than the optimal solution for the MBB problem which partitions the same nodes into $k - 1$ groups, that is $opt_{n,k} \geq opt_{n,k-1}$.*

PROOF. Consider the optimal solution $OPT_{n,k-1} = (B_1, B_2, \dots, B_{k-2})$ which uses exactly $k-1$ groups and add a new border B_{k-1} such that $B_{k-2} < B_{k-1} < n$. Let $SOL_{n,k} = (B_1, B_2, \dots, B_{k-2}, B_{k-1})$ denote such a solution with k groups. By Equation 1, $opt_{n,k-1} = opt_{B_{k-2},k-2} + C_{B_{k-2}+1,n}$. The corresponding cost of $SOL_{n,k}$ is $sol_{n,k} = opt_{B_{k-2},k-2} + (B_{k-1} - B_{k-2})b_{B_{k-1}} + (n - B_{k-1})b_n$, which is no smaller than $opt_{n,k-1} = opt_{B_{k-2},k-2} + (n - B_{k-2})b_n$ because $b_{B_{k-1}} \geq b_n$. This proves that there is a solution with k groups whose cost is no smaller than the optimal solution with $k-1$ groups, and hence $opt_{n,k} \geq sol_{n,k} \geq opt_{n,k-1}$. \square

As a consequence of Lemma 2, the optimal solution with at most K communications always has exactly K communications and its cost is found in $M_{K,N}$.

To evaluate the time complexity of the *DP* algorithm, observe that $O(N)$ comparisons are required to fill every entry of the matrix M , which implies that $O(N^2)$ comparisons are required to fill a row. Since there are K rows, the complexity of the *DP* algorithm is $O(N^2K)$.

To improve on the time complexity of the *DP* algorithm for the *MBB* problem, the properties of optimal solutions have to be further exploited.

DEFINITION 1. Let $1, 2, \dots, N$ be nodes sorted by non-increasing bandwidth. An optimal solution $OPT_{N,K} = (B_1, B_2, \dots, B_{K-1})$ is called left-most optimal and denoted by $LMO_{N,K}$ if it holds $B_{K-1} \leq B'_{K-1}$ for any other optimal solution $(B'_1, B'_2, \dots, B'_{K-1})$.

Clearly, since the problem always admits an optimal solution, there is always a left-most optimal solution. Although the left-most optimal solutions do not need to be unique, it is easy to check that there exists a unique $(B_1, B_2, \dots, B_{K-1})$ such that (B_1, B_2, \dots, B_i) is a left-most optimal solution for partitioning into $i+1$ groups the nodes $1, 2, \dots, B_{i+1}$, for every $i < K$.

DEFINITION 2. A left-most optimal solution $(B_1, B_2, \dots, B_{K-1})$ is called strict left-most optimal solution, and denoted by $SLMO_{N,K}$, if (B_1, B_2, \dots, B_i) is a $LMO_{B_{i+1},i+1}$ for every $i < K$.

The algorithm to be presented will compute a left-most optimal solution for every $i < K$, and thus it will find the unique strict left-most optimal solution.

LEMMA 3. Let the nodes $1, 2, \dots, N$ be sorted by non-increasing bandwidth. Let $LMO_{N-1,K} = (B_1, B_2, \dots, B_{K-1})$ and $OPT_{N,K} = (B'_1, B'_2, \dots, B'_{K-1})$. Then, $B'_{K-1} \geq B_{K-1}$.

PROOF. Let the costs of $LMO_{N-1,K}$ and $OPT_{N,K}$ be, respectively, $opt_{N-1,K} = opt_{B_{K-1},K-1} + (N-1-B_{K-1})b_{N-1}$ and $opt_{N,K} = opt_{B'_{K-1},K-1} + (N-B'_{K-1})b_N$. By contradiction, assume that $B'_{K-1} = B_{K-1} - 1$. We prove that if the optimal solution for N nodes is achieved by moving the last

border towards left, then $LMO_{N-1,K}$ cannot be the optimal solution for the first $N-1$ nodes.

Consider the feasible solution $SOL_{N,K}$ for partitioning N nodes into K communications obtained from $(B_1, B_2, \dots, B_{K-1})$ just adding node N to the K -th communication. Let $sol_{N,K} = opt_{B_{K-1},K-1} + (N-B_{K-1})b_N$ be the cost of such a solution. By the optimality of $OPT_{N,K}$, we have: $opt_{N,K} = opt_{B'_{K-1},K-1} + (N-B'_{K-1})b_N \geq sol_{N,K} = opt_{B_{K-1},K-1} + (N-B_{K-1})b_N = opt_{B_{K-1},K-1} + (N-B'_{K-1}-1)b_N$, or

$$opt_{B'_{K-1},K-1} + b_N \geq opt_{B_{K-1},K-1} \quad (2)$$

Consider now the feasible solution $F_{N-1,K} = (B'_1, \dots, B'_{K-1})$ obtained by truncating the solution $OPT_{N,K}$ to node $N-1$ along with the optimal solution $LMO_{N-1,K}$. Let $opt_{N-1,K}$ and $sol_{N-1,K}$ be the costs associated to $LMO_{N-1,K}$ and $F_{N-1,K}$, respectively.

By the optimality of $LMO_{N-1,K}$, it follows that:

$$\begin{aligned} sol_{N-1,K} &= opt_{B'_{K-1},K-1} + (N-1-B'_{K-1})b_{N-1} \\ &< opt_{N-1,K} = opt_{B_{K-1},K-1} + (N-1-B_{K-1})b_{N-1}, \end{aligned}$$

that is

$$\begin{aligned} opt_{B'_{K-1},K-1} + (N-1-B'_{K-1})b_{N-1} &< opt_{N-1,K} \\ &= opt_{B_{K-1},K-1} + (N-2-B'_{K-1})b_{N-1}, \end{aligned}$$

or

$$opt_{B'_{K-1},K-1} + b_{N-1} < opt_{B_{K-1},K-1}.$$

Recalling that $b_N \leq b_{N-1}$, it follows: $opt_{B'_{K-1},K-1} + b_N \leq opt_{B'_{K-1},K-1} + b_{N-1} < opt_{B_{K-1},K-1}$, contradicting Equation 2. \square

In words, Lemma 3 implies that, given the nodes sorted by non-increasing bandwidths, if one builds an optimal solution for N nodes from an optimal solution for $N-1$ nodes, then the final border B_{K-1} can only move to the right. Such a property can be easily generalized as follows to problems of increasing sizes. From now on, let B_j^i denote the j -th border of $LMO_{i,k}$, with $k > j \geq 1$.

COROLLARY 1. Let the nodes $1, 2, \dots, N$ be sorted by non-increasing bandwidths, and let $l < c < r \leq N$. Then, $B_{K-1}^l \leq B_{K-1}^c \leq B_{K-1}^r$.

PROOF. Follows directly from Lemma 3. \square

In the following we prove that, given n nodes with $1 \leq n \leq N$, if an optimal solution for K transmissions is built from an optimal solution for $K-1$ transmissions, then the final border of the optimal solutions cannot move to the left. Formally, let $LMO_{N,K-1} = (B_1, B_2, \dots, B_{K-2})$ and $OPT_{N,K} = (B'_1, B'_2, \dots, B'_{K-1})$. Then, $B'_{K-1} \geq B_{K-2}$.

Since from now on we can compare solutions with different number n of nodes as well as different number k of transmissions, a not ambiguous notation for the j -th border of

$LMO_{n,k}$ is $B_j^{n,k}$. However, to simplify notation, since in our analysis we always refer to the *last* border of solutions (i.e., $j = k - 1$), to denote the last border of $SOL_{n,k}$ we simply write B_{k-1}^n instead of $B_{k-1}^{n,k}$.

Moreover, let $\delta(n, k) = opt_{n,k} - opt_{n,k-1}$, $1 \leq n \leq N$ and $2 \leq k \leq K$, be the difference of the costs of the two optimal solutions $OPT_{n,K}$ and $OPT_{n,K-1}$.

LEMMA 4. *Let the nodes $1, 2, \dots, N$ be sorted by non-increasing bandwidth. When $K = 2$, $0 \leq \delta(n, 2) \leq \delta(n+1, 2)$ for $1 \leq n \leq N - 1$.*

PROOF. By definition,

$$\begin{aligned} \delta(n, 2) &= opt_{n,2} - opt_{n,1} \\ &= opt_{B_1^n,1} + (n - B_1^n)b_n - nb_n \\ &= opt_{B_1^n,1} - B_1^n b_n \\ &= B_1^n b_{B_1^n} - B_1^n b_n. \end{aligned}$$

Recalling that $B_1^{n+1} \geq B_1^n$ by Lemma 3 and that MBB is a maximization problem (i.e. the optimal solution has a cost no smaller than any feasible solution), $opt_{n+1,2} = opt_{B_1^{n+1},1} + (n+1 - B_1^{n+1})b_{n+1} \geq opt_{B_1^n,1} + (n+1 - B_1^n)b_{n+1}$. Thus,

$$\begin{aligned} \delta(n+1, 2) &= opt_{n+1,2} - opt_{n+1,1} \\ &\geq opt_{B_1^n,1} + (n+1 - B_1^n)b_{n+1} - (n+1)b_{n+1} \\ &= opt_{B_1^n,1} - B_1^n b_{n+1} \\ &= B_1^n b_{B_1^n} - B_1^n b_{n+1} \geq B_1^n b_{B_1^n} - B_1^n b_n = \delta(n, 2). \end{aligned}$$

The claim holds recalling that, by Lemma2, $\delta(n, k) \geq 0$. \square

Moreover:

LEMMA 5. *For fixed k and n , with $2 \leq k \leq K - 1$ and $1 \leq n \leq N$, the final borders of $OPT_{n,k+1}$ and $LMO_{n,k}$, satisfy $B_k^n \geq B_{k-1}^n$ if and only if $\delta(1, k) \leq \delta(2, k) \leq \dots \leq \delta(N, k)$.*

PROOF. Let us start proving that if $\delta(1, k) \leq \delta(2, k) \leq \dots \leq \delta(N, k)$, then $B_k^n \geq B_{k-1}^n$. By contradiction, let us assume

$$B_k^n < B_{k-1}^n.$$

By the optimality of $OPT_{n,k+1}$, it holds:

$$\begin{aligned} opt_{n,k+1} &= opt_{B_k^n,k} + (n - B_k^n)b_n \\ &> sol_{n,k+1} = opt_{B_{k-1}^n,k} + (n - B_{k-1}^n)b_n. \end{aligned}$$

Subtracting $\delta(B_k^n, k)$ and $\delta(B_{k-1}^n, k)$ respectively from $opt_{n,k+1}$ and $sol_{n,k+1}$, one has:

$$opt_{B_k^n,k} + (n - B_k^n)b_n - (opt_{B_k^n,k} - opt_{B_k^n,k-1}) >$$

$$opt_{B_{k-1}^n,k} + (n - B_{k-1}^n)b_n - (opt_{B_{k-1}^n,k} - opt_{B_{k-1}^n,k-1}),$$

since by hypothesis $\delta(B_k^n, k) \leq \delta(B_{k-1}^n, k)$. Thus:

$$(n - B_k^n)b_n + opt_{B_k^n,k-1} > (n - B_{k-1}^n)b_n + opt_{B_{k-1}^n,k-1},$$

or, equivalently:

$$sol_{n,k} > opt_{n,k},$$

raising a contradiction.

On the other side, let $B_k^n \geq B_{k-1}^n$. By the optimality of $OPT_{n,k+1}$ and $OPT_{n,k}$, one has:

$$\begin{aligned} opt_{n,k+1} &= opt_{B_k^n,k} + (n - B_k^n)b_n \geq \\ sol_{n,k+1} &= opt_{B_{k-1}^n,k} + (n - B_{k-1}^n)b_n, \end{aligned}$$

and

$$\begin{aligned} opt_{n,k} &= opt_{B_{k-1}^n,k-1} + (n - B_{k-1}^n)b_n \geq \\ sol_{n,k} &= opt_{B_k^n,k-1} + (n - B_k^n)b_n. \end{aligned}$$

Summing side by side, that is,

$$opt_{n,k+1} + opt_{n,k} \geq sol_{n,k+1} + sol_{n,k},$$

it holds:

$$opt_{B_k^n,k} + opt_{B_{k-1}^n,k-1} \geq opt_{B_{k-1}^n,k} + opt_{B_k^n,k-1},$$

which is equivalent to:

$$opt_{B_k^n,k} - opt_{B_k^n,k-1} \geq opt_{B_{k-1}^n,k} - opt_{B_{k-1}^n,k-1}.$$

Therefore, if $B_k^n \geq B_{k-1}^n$, it must be $\delta_{B_k^n,k} \geq \delta_{B_{k-1}^n,k}$. \square

Given the result in Lemma 4 and applying Lemma 5 when $k = 2$, it yields:

COROLLARY 2. *For each $1 \leq n \leq N$, the final borders B_2^n and B_1^n , respectively, of $OPT(n, 3)$ and $LMO(n, 2)$ satisfy $B_2^n \geq B_1^n$, for any $n \in [1, \dots, N]$. \square*

Now, it remains to be proved that if for a given $2 \leq k \leq K - 1$ the final borders of $OPT_{n,k+1}$ are greater than those of $LMO_{n,k}$, $1 \leq n \leq N$, then the cost differences $\delta(n, k+1) = opt_{n,k+1} - opt_{n,k}$ are increasing with n .

For this purpose, let $\left[\begin{smallmatrix} r \\ w \end{smallmatrix} \right]_{n,k+1}$ denote the difference between the costs of two solutions $SOL_{n,k+1}$ and $SOL_{n,k}$ with last borders in w and r , respectively. Formally, let $SOL_{n,k+1} = (B_1^n, \dots, B_k^n = w)$ and $SOL_{n,k} = (B_1^n, \dots, B_{k-1}^n = r)$, we define $\left[\begin{smallmatrix} r \\ w \end{smallmatrix} \right]_{n,k+1} = sol_{n,k+1} - sol_{n,k}$. Note that when $SOL_{n,k+1} = OPT_{n,k+1}$ and $SOL_{n,k} = OPT_{n,k}$, it holds $\delta(n, k+1) = \left[\begin{smallmatrix} r \\ w \end{smallmatrix} \right]_{n,k+1}$.

LEMMA 6. *For each $1 \leq n \leq N$, consider the final borders B_{k-1}^n and B_k^n of $LMO_{n,k}$ and $OPT_{n,k+1}$, respectively. If $B_k^n \geq B_{k-1}^n$, then $\delta(1, k+1) \leq \delta(2, k+1) \leq \dots \leq \delta(N, k+1)$.*

PROOF. First of all, if $B_k^n \geq B_{k-1}^n$ then $\delta(1, k) \leq \delta(2, k) \leq \dots \leq \delta(N, k)$ by Lemma 5. Fixed n , let $B_k^n = w$, $B_{k-1}^n = r$, $B_k^{n+1} = \bar{\ell}$, and $B_{k-1}^{n+1} = \ell$ to simplify notation. There are only two possible sorted sequence of borders r, w, ℓ and $\bar{\ell}$ because, by Lemma 3, $r \leq \ell$ and $w \leq \bar{\ell}$, and by hypothesis

$r \leq w$ and $\ell \leq \bar{\ell}$. Precisely, either $r \leq w \leq \ell \leq \bar{\ell}$, if $w \leq \ell$, or $r \leq \ell \leq w \leq \bar{\ell}$, if $w \geq \ell$.

In the former case, i.e., $w \leq \ell$, since $opt_{n,k} = opt_{r,k-1} + (n-r)b_n \geq sol_{n,k} = opt_{w,k-1} + (n-w)b_n$, it holds:

$$\begin{aligned} \delta(n, k+1) &= \begin{bmatrix} r \\ w \end{bmatrix}_{n,k+1} \\ &= opt_{n,k+1} - opt_{n,k} \\ &= opt_{w,k} + (n-w)b_n - opt_{r,k-1} - (n-r)b_n \\ &\leq opt_{w,k} - opt_{w,k-1} = \delta(w, k) \end{aligned}$$

Moreover,

$$\begin{aligned} \delta(n+1, k+1) &= \begin{bmatrix} \ell \\ \bar{\ell} \end{bmatrix}_{n+1,k+1} \\ &= opt_{n+1,k+1} - opt_{n+1,k} \\ &= opt_{\bar{\ell},k+1} + (n-1-\bar{\ell})b_{n+1} - opt_{\ell,k} + (n+1-\ell)b_{n+1} \\ &\geq opt_{\ell,k+1} + (n-1-\ell)b_{n+1} - opt_{\ell,k} + (n+1-\ell)b_{n+1} \\ &= opt_{\ell,k+1} - opt_{\ell,k} \\ &\quad \delta(\ell, k) \end{aligned}$$

Recalling that, by hypothesis, $\delta(w, k) \leq \delta(\ell, k)$ when $w \leq \ell$, it yields:

$$\delta(n, k+1) \leq \delta(w, k) \leq \delta(\ell, k) \leq \delta(n+1, k+1).$$

In the latter case, i.e., $w \geq \ell$, observe that:

$$\begin{aligned} \delta(n, k+1) &= \begin{bmatrix} r \\ w \end{bmatrix}_{n,k+1} \\ &= opt_{w,k} + (n-w)b_n - opt_{r,k-1} - (n-r)b_n \\ &\leq opt_{w,k} + (n-w)b_n - (opt_{\ell,k-1} + (n-\ell)b_n) \\ &= \begin{bmatrix} \ell \\ w \end{bmatrix}_{n,k+1} \end{aligned}$$

Moreover,

$$\begin{aligned} \delta(n+1, k+1) &= \begin{bmatrix} \ell \\ \bar{\ell} \end{bmatrix}_{n+1,k+1} \\ &= opt_{\ell,k} + (n+1-\ell)b_{n+1} - opt_{\bar{\ell},k-1} - (n+1-\ell)b_{n+1} \\ &\geq opt_{w,k} + (n+1-w)b_{n+1} - (opt_{\ell,k-1} + (n+1-\ell)b_{n+1}) \\ &= \begin{bmatrix} \ell \\ w \end{bmatrix}_{n+1,k+1} \end{aligned}$$

Finally, recalling that the nodes are sorted by non-increasing bandwidth. i.e., $b_n \geq b_{n+1}$, and $w \geq \ell$, one has:

$$\begin{aligned} &\begin{bmatrix} \ell \\ w \end{bmatrix}_{n+1,k+1} - \begin{bmatrix} \ell \\ w \end{bmatrix}_{n,k+1} = \\ &opt_{w,k} + (n+1-w)b_{n+1} - opt_{\ell,k-1} - (n+1-\ell)b_{n+1} - \\ &\quad opt_{w,k} + (n-w)b_n - opt_{\ell,k-1} + (n-\ell)b_n \\ &= (n+1-w)b_{n+1} - (n+1-\ell)b_{n+1} - (n-\ell)b_n + (n-w)b_n \\ &= (n-w)(b_{n+1} - b_n) - (n-\ell)(b_{n+1} - b_n) = \\ &\quad = (\ell-w)(b_{n+1} - b_n) \geq 0 \end{aligned}$$

Thus, concluding:

$$\begin{aligned} \delta(n, k+1) &= \begin{bmatrix} r \\ w \end{bmatrix}_{n,k+1} \leq \begin{bmatrix} \ell \\ w \end{bmatrix}_{n,k+1} \leq \\ &\begin{bmatrix} \ell \\ w \end{bmatrix}_{n+1,k+1} \leq \begin{bmatrix} \ell \\ \bar{\ell} \end{bmatrix}_{n+1,k+1} = \delta(n+1, k+1) \quad \square \end{aligned}$$

In conclusion:

LEMMA 7. *Let the nodes $1, 2, \dots, N$ be sorted by non-increasing bandwidth. For each $2 \leq k \leq K$, $\delta(1, k) \leq \delta(2, k) \leq \dots \leq \delta(N, k)$.*

PROOF. The statement holds for $k = 2$ by Lemma 4. By induction on k , we first prove the claim for $k = 3$. Since by Corollary 2, the final borders B_2^n and B_1^n of, respectively, $LMO_{n,2}$ and $OPT_{n,3}$ satisfy $B_2^n \geq B_1^n$, for any $n \in [1, \dots, N]$, then, by applying Lemma 6, one obtains $\delta(1, 3) \leq \delta(2, 3) \leq \dots \leq \delta(N, 3)$.

Now, assume by inductive hypothesis that the theorem holds up to transmission \bar{k} , that is, $\delta(1, \bar{k}) \leq \delta(2, \bar{k}) \leq \dots \leq \delta(N, \bar{k})$ for $3 \leq \bar{k} \leq K-1$. By Lemma 5, the final borders $B_{\bar{k}}^n$ and $B_{\bar{k}-1}^n$ of $OPT_{n, \bar{k}+1}$ and $LMO_{n, \bar{k}}$, respectively, satisfy $B_{\bar{k}}^n \geq B_{\bar{k}-1}^n$. Applying then Lemma 6, we obtain $\delta(1, \bar{k}+1) \leq \delta(2, \bar{k}+1) \leq \dots \leq \delta(N, \bar{k}+1)$ as desired. \square

Therefore:

THEOREM 1. *Let the nodes $1, 2, \dots, N$ be sorted by non-increasing bandwidth. Let $LMO_{N, K-1} = (B_1, B_2, \dots, B_{K-2})$, $LMO_{N-1, K} = (\bar{B}_1, \bar{B}_2, \dots, \bar{B}_{K-1})$, and $OPT_{N, K} = (B'_1, B'_2, \dots, B'_{K-1})$. Then, $B'_{K-1} \geq \max\{B_{K-2}, \bar{B}_{K-1}\}$.*

PROOF. By Lemma 7, it holds $\delta(1, K-1) \leq \delta(2, K-1) \leq \dots \leq \delta(N, K-1)$. Hence, by Lemma 5, $B'_{K-1} \geq B_{K-2}$. Moreover, by Lemma 3, $B'_{K-1} \geq \bar{B}_{K-1}$. Thus, $B'_{K-1} \geq \max\{B_{K-2}, \bar{B}_{K-1}\}$. \square

To speed up the DP algorithm, assume that $LMO_{n, k-1}$ has been found for every $n \in [1, \dots, N]$. If the $LMO_{l, k}$ and $LMO_{r, k}$ solutions are also known for some $1 \leq l \leq r \leq N$, then by Theorem 1, Corollary 1 and Lemma 2, one knows that B_{k-1}^c is between $\max\{B_{k-2}^c, B_{k-1}^l\}$ and $\min\{c-1, B_{k-1}^r\}$, for any $l \leq c \leq r$. Thus, the recurrence in Equation 1 can be rewritten as:

$$opt_{c,k} = \max_{\ell \in \{L, \dots, R\}} \{opt_{\ell, k-1} + C_{\ell+1, c}\} \quad (3)$$

where $L = \max\{B_{k-2}^c, B_{k-1}^l\}$ and $R = \min\{c-1, B_{k-1}^r\}$.

An improved $O(KN \log N)$ time algorithm for the *MBB* problem can be derived by choosing $c = \lceil \frac{L+R}{2} \rceil$ in the recurrence of Equation 3.

The recurrence in Equation 3 is the very similar to Equation 6 of [1], except that maximization replaces minimization and that the $C_{i,h}$'s are defined in a different way. Therefore, it

<i>Input:</i>	N nodes sorted by non-increasing bandwidths, and K communications;
<i>Initialize:</i>	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 $M_{k,i} \leftarrow -\infty$;
<i>Loop 1:</i>	for k from 2 to K do $F_{k,0} \leftarrow F_{k,1} \leftarrow 1$; $F_{k,N+1} \leftarrow N$;
<i>Loop 2:</i>	for t from 1 to $\lceil \log N \rceil$ do
<i>Loop 3:</i>	for i from 1 to 2^{t-1} do $j \leftarrow \lceil \frac{2i-1}{2^t}(N+1) \rceil$; $l \leftarrow \lceil \frac{i-1}{2^{t-1}}(N+1) \rceil$; $r \leftarrow \lceil \frac{i}{2^{t-1}}(N+1) \rceil$;
<i>Loop 4:</i>	if $M_{k,j} = -\infty$ then for ℓ from $\max\{F_{k-1,j}; F_{k,l}\}$ to $\min\{j-1; F_{k,r}\}$ do if $M_{k-1,\ell} + C_{\ell+1,j} > M_{k,j}$ then $M_{k,j} \leftarrow M_{k-1,\ell} + C_{\ell+1,j}$; $F_{k,j} \leftarrow \ell$;

Figure 1: The MBB-Dichotomic algorithm.

can be solved by adapting the *Dichotomic* algorithm presented in [1] to the *MBB* problem.

The MBB-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 j 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 j which have been already filled, with $l < j < r$.

If no entry before j has been already filled, then $l = 1$, and therefore the final border $F_{k,1}$ is initialized to 1. If no entry after j has been filled, then $r = N$, and thus the final border $F_{k,N+1}$ is initialized to N . To compute the entry j , the variable ℓ takes all values between $\max\{F_{k-1,j}; F_{k,l}\}$ and $\min\{j-1; F_{k,r}\}$. The index ℓ which maximizes the recurrence in Loop 4 is assigned to $F_{k,j}$, while the corresponding maximum value is assigned to $M_{k,j}$.

To show the correctness, consider how a generic row k is filled up. In the first stage (i.e. $t = 1$), the entry $M_{k,\lceil \frac{N+1}{2} \rceil}$ is filled and ℓ ranges over almost all values $1, \dots, N$. By Corollary 1, observe that to fill an entry $M_{k,l}$ where $l < \lceil \frac{N+1}{2} \rceil$, one needs to consider at most all the entries $M_{k-1,\ell}$ where $\ell \leq F_{k,\lceil \frac{N+1}{2} \rceil}$. Similarly, to fill an entry $M_{k,l}$ where $l > \lceil \frac{N+1}{2} \rceil$, one needs to consider the entries $M_{k-1,\ell}$ where $\ell \geq F_{k,\lceil \frac{N+1}{2} \rceil}$. In general, one can show that in stage t , to compute the entries $M_{k,j}$ with $j = \lceil \frac{2i-1}{2^t}(N+1) \rceil$ and $1 \leq i \leq 2^{t-1}$, at most the entries $M_{k-1,\ell}$ must be considered, where $F_{k,l} \leq \ell \leq F_{k,r}$ and l and r are $\lceil \frac{i-1}{2^{t-1}}(N+1) \rceil$ and $\lceil \frac{i}{2^{t-1}}(N+1) \rceil$, respectively. Notice that these entries have been computed in earlier stages. The above process repeats for every row of the matrix. The algorithm proceeds until the last entry $M_{K,N}$ is computed. The strict left-most optimal solution $SLMON,K = (B_1, B_2, \dots, B_{K-1})$ is obtained, where $B_{k-1} = F_{k,B_k}$ for $1 < k \leq K$ and $B_K = N$.

LEMMA 8. *The total number of comparisons involved in a stage of the MBB-Dichotomic algorithm is $O(N)$.*

PROOF. The whole execution of Loop 3 of Figure 1 cor-

responds to the execution of a stage for a particular value of t . The total number of comparisons involved is equal to the sum of the number of values the variable ℓ takes in Loop 3, which is

$$\leq \sum_{i=1}^{2^{t-1}} (F_{k,r} - F_{k,l} + 1) \quad \text{where } l = \left\lceil \frac{i-1}{2^{t-1}}(N+1) \right\rceil$$

$$\quad \text{and } r = \left\lceil \frac{i}{2^{t-1}}(N+1) \right\rceil$$

Therefore:

$$\begin{aligned} & \sum_{i=1}^{2^{t-1}} (F_{k,\lceil \frac{i}{2^{t-1}}(N+1) \rceil} - F_{k,\lceil \frac{i-1}{2^{t-1}}(N+1) \rceil} + 1) = \\ & \quad (F_{k,\lceil \frac{N+1}{2} \rceil} - F_{k,0} + 1) + \\ & \quad (F_{k,\lceil \frac{2}{2^{t-1}}(N+1) \rceil} - F_{k,\lceil \frac{N+1}{2} \rceil} + 1) + \\ & \quad \vdots \\ & \quad + (F_{k,\lceil \frac{2^{t-1}}{2^{t-1}}(N+1) \rceil} - F_{k,\lceil \frac{2^{t-1}-1}{2^{t-1}}(N+1) \rceil} + 1) = \\ & \quad F_{k,N+1} - F_{k,0} + 2^{t-1} = \\ & \quad N - 1 + 2^{t-1} = \\ & \quad O(N) \quad \square \end{aligned}$$

THEOREM 2. The *MBB* Problem can be solved by the MBB-Dichotomic algorithm in $O(KN \log N)$ time.

PROOF. From Lemma 8, one stage of Figure 1, corresponding to the execution of Loop 2 for a particular value of t , involves $O(N)$ comparisons. Since Loop 2 runs $\lceil \log N \rceil$ times and Loop 1 is repeated K times, the overall time complexity is $O(NK \log N)$. \square

3. MULTIPLE INTERFACES

Let S be a set of N nodes and I be a set of H interfaces. Each node j is associated with a subset of interfaces I_j . Let $b_{i,j}$ be the maximum bandwidth that can be used by

a communication toward node j by means of interface i . If $i \notin I_j$ then $b_{i,j} = 0$. Hence, for the ease of notation, by properly setting the values $b_{i,j}$ we can always assume that all nodes hold all the interfaces. A transmission that simultaneously serves a subset of nodes $A_i \subseteq S$ by means of interface i is assumed to transfer data at a rate equal to $\min_{j \in A_i} \{b_{i,j}\}$. The *K-Maximum Bandwidth Broadcast in Multi-Interface Networks* (*MBBM* for short) can be stated as follows.

MBBM: K-Maximum Bandwidth Broadcast in Multi-Interface Networks

Input: A set S of N nodes and a set I of H interfaces. A bandwidth function $b : S \times I \rightarrow \mathbb{R}_0^+$ and an integer $K \leq H$.

Solution: A subset I' of K interfaces and a partition A_1, A_2, \dots, A_K of S which associates each subset of nodes to one different interface in I' .

Goal: Maximize $\sum_{i \in I'} |A_i| \min_{j \in A_i} \{b_{i,j}\}$.

In other words, we want to determine which subset of interfaces and, for each interface, which subset of nodes the source has to reach by means of a specific interface in order to maximize the bandwidth of communication during a broadcast transmission, assuming that each node cannot be reached simultaneously by means of different interfaces.

The *MBBM* problem can be also visualized by considering a matrix D with H rows, one for each interface, and N columns, one for each node, where each entry $D[i, j] = b_{i,j}$. Then, the solution provides the selection of the subset I' of K rows, and for each selected row $i \in I'$, the selection of some columns A_i in such a way that each column is associated with one and only one row.

In the multiple interface case, we introduce the generalized *multi-interface segmentation* in order to characterize optimal solutions for the *MBBM* problem.

Let $I' = \{r^1, r^2, \dots, r^K\}$ be a subset of K rows and let A_{r^1}, \dots, A_{r^K} be a partition of the N columns of D , where A_{r^i} denotes the subset of columns assigned to interface r^i , with $1 \leq i \leq K$. Moreover, let the K rows r^1, r^2, \dots, r^K be indexed according to the induced minimum values of the bandwidths in the corresponding A_{r^i} columns obtaining that $r_{min}^1 \geq r_{min}^2 \geq \dots \geq r_{min}^K$. In a *multi-interface segmentation*, the columns of group A_{r^i} of D served by interface r^i form a consecutive subset of columns in row r^i once the columns in $A_{r^1}, \dots, A_{r^{i-1}}$ have been canceled from D and the remaining columns $S - (A_{r^1} \cup A_{r^2} \cup \dots \cup A_{r^{i-1}})$ in r^i have been sorted in non-increasing bandwidth order. In other words, in a multi-interface segmentation, if columns x and y belong to the group of columns assigned to interface r^i and $D[r^i, x] < D[r^i, y]$, then all the columns z in row r^i not yet assigned to interfaces r^1, \dots, r^{i-1} with bandwidth $D[r^i, x] \leq D[r^i, z] \leq D[r^i, y]$ are also associated to the same interface r^i .

LEMMA 9. *The optimal solution for MBBM is a multi-interface segmentation.*

PROOF. Given an optimal solution using K interfaces, we can order the K out of H chosen rows r^1, r^2, \dots, r^K according to the induced minimum values of the bandwidths in the corresponding columns, obtaining that $r_{min}^1 \geq r_{min}^2 \geq \dots \geq r_{min}^K$. Sorted row r^1 in non-increasing order and indexed the columns of D according to such an order, let us assume by contradiction that A_{r^1} consists of several distinct consecutive intervals of columns in D . All the intervals of columns not associated to r^1 before the last interval that contains $D[r^1, n_1] = r_{min}^1$ can be assigned to r^1 without affecting the solution because each column in such intervals contributes to the optimal solution with a value no greater than r_{min}^1 .

After filling all the first intervals on r^1 , we have $A_{r^1} = [1..n_1]$. Then, we can consider row r^2 excluding the columns of D already assigned to r^1 . Sorted row r^2 in non-increasing order and indexed the columns $S - A_{r^1}$ of D according to such an order, let us assume by contradiction that A_{r^2} consists of distinct intervals of columns of D . As before, each column z not associated to r^2 on the left of column n_2 such that $D[r^2, n_2] = r_{min}^2$ can be moved in A_{r^2} without decreasing the cost of the solution because they contribute at most r_{min}^2 to the optimal solution.

Repeating the same until the last row r^K is considered or until all the N columns are assigned, we obtain a multi-interface segmentation that provides at least the same bandwidth of the original optimal solution. Hence there is an optimal solution which is a multi-interface segmentation. \square

Although the optimal solutions of *MBBM* still satisfy a kind of segmentation, the major difficulty consists in choosing the K rows along with their permutation that lead to the optimal solution. Thus, introducing multiple interfaces makes the problem much harder.

3.1 Complexity

In this section we study the complexity of *MBBM* and we prove that the problem is computationally intractable.

THEOREM 3. *MBBM is NP-hard.*

PROOF. We prove that the underlying decisional problem, denoted by *MBBM_D*, is in general *NP*-complete. We need to add one bound $B \in \mathbb{R}_0^+$ such that the problem will be to ask whether there exists a partition of S composed of K subsets which induces a minimum bandwidth of transmission of at least B .

The problem is in *NP*. In fact, given a partition for an instance of *MBBM_D*, checking whether it ensures a bandwidth of at least B requires linear time in the size of the instance.

The proof then proceeds by means of a polynomial reduction from the well-known *Exact Cover by 3-Sets* problem. Such a problem is known to be *NP*-complete [2] and it can be stated as follows:

Input: Set X with $|X| = 3q$ and a collection C of 3-element subsets of X .

Question: Is there an exact set cover for X , i.e. a subset $C' \subseteq C$ such that $|C'| = q$ and every element of X belongs to exactly one member of C' ?

Given an instance of $X3C$, we build an instance of $MBBM_D$ in polynomial time as follows. Let $K = q$, the set of nodes S of $MBBM_D$ is composed of $N = |X| = 3q$ nodes and $|I| = H = |C|$. Hence, we define two mappings. One is between X and S , the other is between subsets C and interfaces I . For each subset $c \in C$, if an element x corresponding to node j belongs to c which corresponds to an interface i , then $b_{i,j} = 1$, otherwise $b_{i,j} = 0$. It follows that each interface can be used to reach at most 3 nodes while it guarantees one unit of bandwidth for each node. Finally, let $B = 3q$. We need to prove that a solution to $X3C$ is possible if and only if there exists a solution to the corresponding instance of $MBBM_D$.

(\Rightarrow): Let us suppose that $X3C$ admits a solution. The covering must consist of q triples. From the $MBBM_D$ perspective, the q triples correspond to K subsets A_1, A_2, \dots, A_K chosen to transmit data to the $3q$ nodes. Each subset corresponds to one different interface. By construction, for each $j \in S$ there is a unique subset A_i in the induced solution of $MBBM_D$ such that $b_{i,j} = 1$, hence A_1, A_2, \dots, A_K represent a partition of S . Summing up over all the bandwidth allowed by the corresponding interfaces, we obtain: $\sum_{i=1}^K |A_i| \min_{j \in A_i} \{b_{i,j}\} = \sum_{i=1}^K 3 = 3K = 3q = B$.

(\Leftarrow): Let us suppose that $MBBM_D$ admits a solution. By construction, each of the K chosen interfaces can be used to transmit one unit of bandwidth to at most 3 nodes. Since we have $K = q$ and $B = 3q$, each interface must necessarily be used to transmit one unit of bandwidth to 3 nodes. Hence, the partition of S provided by the assumed solution corresponds to a set of K triples in $X3C$ that covers all the elements in X , that is $X3C$ admits a solution. \square

3.2 Particular cases

In this section, some special cases are considered where the $MBBM$ problem can be efficiently solved. The following theorem can be stated.

THEOREM 4. *If $K \leq 2$, then $MBBM$ is polynomially solvable.*

PROOF. If $K = 1$, it is easy to check which row of D admits the highest minimum among all the columns, and this clearly determines the selection of the best row. Overall, $O(NH)$ time is required.

If $K = 2$, we may consider every pair of rows. For each pair, let us sort all the columns according to the non-increasing order of the first chosen row. Once chosen the right couple of rows and the right order (either with respect to the first chosen row or to the second one), we only need to find the best border B_1 among N possibilities by Lemma 9.

The overall complexity of the above described algorithm is $O(NH(\log N + H))$. Indeed, one can choose one of the H rows at a time, order the columns in $O(N \log N)$ time according to the just chosen row, consider all the $H - 1$ pairs of rows consisting of the just chosen row and every other row, and find in $O(N)$ time the best border B_1 , for a total of $O(H(N \log N + NH))$ time. \square

Consider now the particular case when there is a way of indexing the columns of D that simultaneously sort all the rows of D . In other words, arranged the columns of D in such a way that a given row is sorted in non-increasing order, all the rows of D are sorted in non-increasing order of their bandwidths. When this property holds we say that the instance respects a *common order*. From now on, let $CMBBM$ denote the $MBBM$ problem when the common order holds. In $CMBBM$, we assume the rows of D indexed so that all the rows are sorted. In practice, the $CMBBM$ problem might arise when the bandwidth achievable by a node with any interface depends, in the same way, on the distance of the node from the source s .

When the common order holds, a multi-interface segmentation becomes a partition A_1, \dots, A_K of the columns of D where each group A_i belongs to a different row (i.e., it is assigned to a different interface) and consists of consecutive columns. In such a case a multi-interface segmentation can be denoted by the K -tuple

$$\langle (B_1; i_1), (B_2; i_2), \dots, (B_{K-1}; i_{K-1}), (N; i_K) \rangle$$

where border B_j is the index of the last column that belongs to group A_j assigned to interface i_j . Note that in this case it is necessary to indicate also the last border $B_K = N$ since we do not know to which interface group A_K is assigned.

To design an optimal enumeration algorithm for $CMBBM$, one should consider all the possible multi-interface segmentations of the columns of matrix D and, for each subset of K rows, find the best solution using at most such rows. An improved enumeration algorithm can be achieved exploiting a reduction to a *Resource Constrained Shortest Paths* (briefly, $RCSP$) problem on directed multigraphs, which is defined as follows [4].

RCSP: Resource Constrained Shortest Paths

Input: A directed multigraph $G = (V, E)$ with two special vertices s and t , an integer K (which is the number of resources) and a (positive integer) resource availability vector (R_1, R_2, \dots, R_K) . Each edge $e \in E$ has a positive weight $w(e)$ and a (positive integer) resource request vector $(r_1(e), r_2(e), \dots, r_K(e))$.

Solution: A feasible path p from s to t such that $\sum_{e \in p} r_i(e) \leq R_i$ for $1 \leq i \leq K$.

Goal: Minimize $\sum_{e \in p} w(e)$ over all the feasible paths p .

To reduce $CMBBM$ to $RCSP$ when the set I consists of K interfaces, define a vertex i for column i of D , with $1 \leq i \leq N$, and add vertex $s = 0$. Let vertex t be equal to N

and the resource availability vector be such that $R_i = 1$, for $1 \leq i \leq K$. For every pair of vertices i, j such that $0 \leq i < j \leq N$ and every $1 \leq k \leq K$ add the edge $e = (i, j)$ with $w(e) = -(j - i)b_j$ and resource request vector $(r_1(e), r_2(e), \dots, r_K(e))$ with all entries equal to zero except for $r_k = 1$. Clearly, $|V| = O(N)$ and $|E| = O(KN^2)$.

Note that *RCSP* requires positive edge weights, while the weights introduced in the reduction are negative because the *CMBBM* problem is a maximization problem. However, since the so constructed multigraph is acyclic, the *RCSP* problem is well defined even if the edge weights are negative.

A feasible shortest path from s to t represents an optimal solution for the *CMBBM* problem when $H = K$. Note that if an optimum path p has $\sum_{e \in p} r_i(e) = 0$ for some $1 \leq i \leq K$, it means that in the optimum solution interface i is not used.

THEOREM 5. *If all the H rows of matrix D respect a common non-increasing order, *CMBBM* can be solved in polynomial time when*

1. $K = O(1)$, or
2. $H - K = O(1)$ and $K = O(\log^\alpha N)$, where $\alpha = O(1)$.

PROOF. A modified version of Dijkstra's algorithm can be used to solve *RCSP*. Precisely, for each vertex v one has to save into a priority queue not only the weight of the path to reach v from s but also the set of resources used along such a path. Moreover, leaving vertex v , an edge can be used only if its resource has not already been used in the path from s to v . Since each vertex v can be reached with at most $O(2^K)$ different paths, the size of the priority queue can grow as much as $O(2^K N)$. Hence, the edges that have to be relaxed during the entire algorithm are $O(2^K KN^2)$. So, implementing the priority queue with a Fibonacci heap, *MBBM* requires $O(2^K KN^2 + 2^K N \log(2^K N)) = O(2^K KN^2)$ time.

In the general case, when *CMBBM* has $K \leq H \leq N$, *CMBBM* can be solved applying *RCSP* $\binom{H}{K}$ times, once for each subset of K interfaces out of the H available interfaces, for an overall time of $O(\binom{H}{K} 2^K KN^2)$. It is well known [3] that

$$\binom{H}{K} \approx \begin{cases} \frac{H^K}{K!} & \text{if } K \text{ is a constant} \\ \frac{H^{H-K}}{(H-K)!} & \text{if } H - K \text{ is a constant} \end{cases}$$

In the former case, the time complexity $O(\binom{H}{K} 2^K KN^2)$ becomes $O(H^K N^2)$, which is polynomial because $K = O(1)$ and $H \leq N$. In the latter case, if $K = O(\log^\alpha N)$ then $H = O(\log^\alpha N)$ too since $H - K$ is a constant, and thus the complexity is $O(H^{H-K} 2^K KN^2) = O((\log N)^{\alpha(H-K+1)} N^{\alpha+2})$, which is polynomial because $\alpha = O(1)$. \square

It is worthy to note that, for the *CMBBM* problem, if one row of D contains the minimum in each column of D and $H > K$, then such a row of D can be dropped without affecting the optimal solution.

As a further particular case, consider the *CMBBM* problem where each interface can be reused in more than one transmission. Precisely, we allow that two or more groups

of the multi-interface segmentation can be associated with the same row.

Generalizing the definitions given for the single interface case, given $n \leq N$ and $k \leq K$, let $OPT_{n,k}$ denote an optimal solution for grouping nodes $1, \dots, n$ into k groups and let $opt_{n,k}$ be its corresponding cost. Let $C_{i,h;m}$ be the cost of assigning consecutive nodes i, \dots, h to one group using interface m , i.e. $C_{i,h;m} = (h - i + 1)b_{m,h}$. Hence, $opt_{n,1} = \max_{m \in \{1,2,\dots,H\}} C_{1,n;m} = n \max_{m \in \{1,2,\dots,H\}} b_{m,n}$ for every n . For $1 < k \leq K$, the following recurrence holds:

$$opt_{n,k} = \max_{\ell \in \{1,2,\dots,n-1\}} \max_{m \in \{1,2,\dots,H\}} \{opt_{\ell,k-1} + C_{\ell+1,n;m}\} \quad (4)$$

By the recurrence in Equation 4, a dynamic programming algorithm can be readily derived to solve this problem variant in $O(N^2 HK)$ time.

4. CONCLUSION

A broadcasting problem in multi-interface networks has been considered where each transmission, simultaneously serving a group of nodes, has an overall bandwidth equal to the product between the smallest node bandwidth in the group and the cardinality of the group. We introduced the *K-Maximum Bandwidth Broadcast in Single-Interface Networks (MBB)* and *K-Maximum Bandwidth Broadcast in Multiple-Interface Networks (MBBM)* problems. The former problem uses exactly K transmissions to cover all the N nodes and can be optimally solved in $O(NK \log N)$ time. In contrast, the latter problem uses at most K interfaces, one for each transmission, and it is computationally intractable (i.e. NP-hard). However, it becomes polynomially solvable in some particular cases, namely, when either $K \leq 2$ or a common order holds and K is polylogarithmic in N . As a further work, we intend to devise in the future good heuristics for the general *MBBM* problem.

5. REFERENCES

- [1] 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 Trans. Computers*, 54(5):558–572, 2005.
- [2] M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.
- [3] R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics, Second Edition*. Addison-Wesley, Reading, Massachusetts, 1994.
- [4] G. Handler and I. Zang. A dual algorithm for the constrained shortest path problem. *Networks*, 10:293–310, 1980.
- [5] J. Qadir, C.T. Chou, A. Misra, and J.G. Lim. Localized minimum-latency broadcasting in multi-radio multi-rate wireless mesh networks. In *WOWMOM*, pages 1–12, 2008.
- [6] J. Qadir, C.T. Chou, A. Misra, and J.G. Lim. Minimum latency broadcasting in multiradio, multichannel, multirate wireless meshes. *IEEE Trans. Mob. Comput.*, 8(11):1510–1523, 2009.