Broadcasting on Large Scale Heterogeneous Platforms under the Bounded Multi-Port Model

Olivier Beaumont, Lionel Eyraud-Dubois, Shailesh Agrawal Kumar

Broadcasting on Large Scale Heterogeneous Platforms under the Bounded Multi-Port Model

Olivier Beaumont, Lionel Eyraud-Dubois
INRIA Bordeaux – Sud-Ouest
University of Bordeaux, LaBRI
Bordeaux, France
Email: obeaumon@labri.fr, eyraud@labri.fr

Shailesh Kumar Agrawal
INRIA Bordeaux – Sud-Ouest
Bordeaux, France
and
Department of CSE
Indian Institute of Technology Kanpur
Email: sagrawal@cse.iitk.ac.in

***Abstract*—We consider the problem of broadcasting a large message in a large scale distributed platform. The message must be sent from a source node, with the help of the receiving peers which may forward the message to other peers. In this context, we are interested in maximizing the throughput (*i.e.* the maximum streaming rate, once steady state has been reached).**

The platform model does not assume that the topology of the platform is known in advance: we consider an Internet-like network, with complete potential connectivity. Furthermore, the model associates to each node local properties (incoming and outgoing bandwidth), and the goal is to build an overlay which will be used to perform the broadcast operation. We model contentions using the bounded multi-port model: a processor can be involved simultaneously in several communications, provided that its incoming and outgoing bandwidths are not exceeded. For the sake of realism, it is also necessary to bound the number of simultaneous connections that can be opened at a given node (ie its outdegree).

We prove that unfortunately, this additional constraint makes the problem of maximizing the overall throughput NP-Complete. On the other hand, we also propose a polynomial time algorithm to solve this problem, based on a slight resource augmentation on the outdegree of the nodes.

***Keywords*-Broadcast; Scheduling; Resource Augmentation; NP Completeness; Approximation Algorithms; Communication modeling**

I. Introduction

Data dissemination in distributed platforms has been the subject of a vast literature. The problem comes in two flavors, depending on the context. On the one hand, if the topology of the platform is known (in the cas of computer networks or parallel machines for example), the goal is to organize data transfers so as to maximize the throughput (or minimize the makespan for a given message size). On the other hand, in the context of large scale, internet level platforms, the goal is to find the topology (the overlay network) that would maximize the throughput. In this paper, we consider both problems: we provide an efficient overlay and explicit the data transfers.

Broadcasting in computer networks is the focus of a vast literature. The one-to-all broadcast, or single-node broadcast, is the most primary collective communication pattern: initially, only the source processor has the data that needs to be broadcast; at the end, there is a copy of the original data residing at each processor. Parallel algorithms often require to send identical data to all other processors, in order to disseminate global information (typically, input data such as the problem size or application parameters). Numerous broadcast algorithms have been designed for parallel machines such as meshes, hypercubes, and variants (see among others [1], [2], [3]).

The same framework applies for broadcasting a live stream of data, such as a movie. In the context of content distribution systems, it is at the core of live streaming distribution systems such as CoolStreaming [4], PPLive [5] or SplitStream [6]. In this case also, we are interested in the distribution of a large message to all the nodes of a large scale platform. These platforms are made of a large number of computers, geographically distributed, and interconnected by the Internet network. In the context of this work, it is thus not possible to have access to the core of the network, and we are interested in application-level solutions. The goal of this work is to construct an overlay that makes the best possible use of the communication capabilities of all participating nodes, so as to maximize the overall streaming rate (once steady-state has been reached).

In our platform model, we assume (as in the Internet) that each node has the possibility to communicate with any other nodes. All nodes have different speeds of communications, both for incoming and outgoing communication, and we further assume that these are the only points where contention may happen (*i.e.*, that the capacity of the core of the network is large enough). This allows to represent the platform by associating local properties to each node (namely its incoming and outgoing bandwidths), whose values can be determined easily.

To model contentions, we rely on the bounded multi-port model, that has already been advocated by Hong et al. [7] for independent task distribution on heterogeneous platforms. In this model, node $\mathcal{N}_i$ can communicate with any number of nodes $\mathcal{N}_j$ simultaneously, each using a bandwidth $c(\mathcal{N}_i, \mathcal{N}_j)$

provided that its outgoing bandwidth is not exceeded, *i.e.*, $\sum_j c(\mathcal{N}_i, \mathcal{N}_j) \leq b_i^{\text{out}}$. Similarly, node $\mathcal{N}_i$ can receive messages from any number of nodes $\mathcal{N}_j$ simultaneously, each using a bandwidth $c(\mathcal{N}_j, \mathcal{N}_i)$, provided that its incoming bandwidth is not exceeded, *i.e.*, $\sum_j c(\mathcal{N}_j, \mathcal{N}_i) \leq b_i^{\text{in}}$. This corresponds well to modern network infrastructure, where each communication is associated to a TCP connection.

This model strongly differs from the traditional one-port model used in scheduling literature, where connections are made in exclusive mode: each node can communicate with a single node at any time-step. Previous results obtained in steady-state scheduling of broadcasts [8] have been obtained under this model. Saif and Parashar [9] report experimental evidence that achieving the performance of bounded multi-port model may be difficult using standard message passing libraries, since asynchronous sends become serialized as soon as message sizes exceed a few megabytes. Their results hold for two popular implementations of the MPI message-passing standard, MPICH on Linux clusters and IBM MPI on the SP2. Nevertheless, in the context of large scale platforms, the networking heterogeneity ratio may be high, and it is unacceptable to assume that a 100MB/s server may be kept busy for 10 seconds while communicating a 1MB data file to a 100kB/s DSL node. Therefore, in our context, we will assume that all communications are directly handled at TCP level (even bounded multi-port model applies to recent multi-threaded implementations of high level communication libraries such as MPICH 2).

It is worth noting that at the operating system level, several QoS mechanisms enable a prescribed sharing of the bandwidth [10], [11]. In particular, it is possible to handle simultaneously several connections and to fix the bandwidth allocated to each connection. In our context, these mechanisms are particularly useful since the bandwidth allocated to the connection between $\mathcal{N}_j$ and $\mathcal{N}_i$ may be lower than both b_j^{out} and b_i^{in}. Nevertheless, handling a large number of connections at $\mathcal{N}_j$ with prescribed bandwidths consumes many kernel resources, and it may therefore be difficult to reach b_j^{out} by aggregating the bandwidths of a large number of connections. Moreover, the maintenance cost of the overlay network is directly related to the maximum degree of a node, so that large connectivity would induce a high maintenance overhead. In order to avoid this problem, we introduce another parameter d_j in the bounded multi-port model, that represents the maximal number of connections that can simultaneously be opened and handled with QoS mechanisms at node $\mathcal{N}_j$.

Therefore, the model we propose encompasses the benefits of both the bounded multi-port model and the one-port model. It enables several communications to take place simultaneously, what is compulsory in the context of large scale distributed platforms, and practical implementation is achieved by using TCP QoS mechanisms and by bounding the maximal number of connections.

In summary, our goal is both to design an overlay network and to determine the bandwidths associated to the edges of the overlay, such that both degree and capacity constraints are satisfied, and such that the overall throughput that can be reached using this overlay network is close to the optimal one. Then, we also need to provide a broadcast algorithm (*i.e.* a description of what part of the message is sent on each edge) that achieves the claimed throughput.

The rest of the paper is organized as follows. In Section II, we present the communication model we use and formalize the scheduling problem we consider. We prove in Section III that if we introduce a bound on the maximal number of connections that can be opened simultaneously at a node, then the problem of maximizing the overall throughput becomes NP-Complete. In Section IV, we propose a sophisticated polynomial time algorithm, based on a slight resource augmentation on the degree of the nodes, to solve this problem. Section V presents extensive simulation results comparing our approach to greedy based heuristics. We also discuss related works and implementation issues in Section VI. At last, we provide in Section VII some future works and concluding remarks.

II. Problem Modeling

Let us denote by b_j the outgoing bandwidth of node $\mathcal{N}_j$ and by d_j the maximal number of outgoing connections that it can handle simultaneously (its degree). Our aim is to broadcast the same message with throughput T to all the nodes of the platform. Clearly, all nodes need to receive the message exactly once, so that all incoming bandwidths should be at least T. Therefore, in what follows, we assume that the incoming bandwidths of all nodes are larger than T and we concentrate only on outgoing communication capacities.

Let us consider for instance the following platform, depicted in Figure 1. The platform consists in 3 nodes: the source node (the one that initially holds or generate the message), whose outgoing bandwidth is 2 and degree is 2, and 2 receiving nodes $\mathcal{N}_1$ and $\mathcal{N}_2$ having the same characteristics: degree 1 and outgoing bandwidth 1. We depict in Figure 1 one of the best possible trees ($T = 1$) to broadcast the message, one of the best possible DAG (Directed Acyclic Graph) with $T = 1.5$ and the best possible solution ($T = 2$), that consists in using 2 trees, each tree providing $T = 1$.

This example shows that, in order to optimize the throughput, we have to look for a solution made of several weighted trees that will be used to broadcast the different parts of the message to all the nodes. More precisely, we can formalize the problem as follows:

BcastTrees: Let b_j and d_j denote the outgoing bandwidth and outdegree of $\mathcal{N}_j$ (the source node being denoted as $\mathcal{N}_0$). Find a set of directed spanning trees T_k rooted at $\mathcal{N}_0$, and associated weights $w_k > 0$, such that $\sum_k w_k$ is

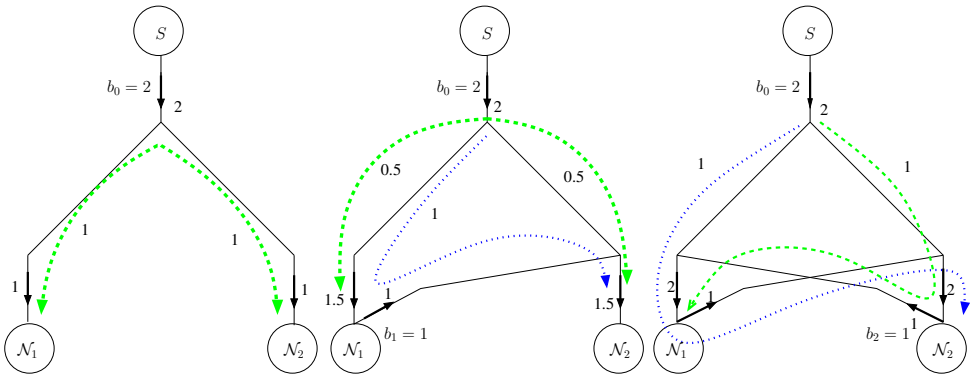


Figure 1. Example with an optimal tree, an optimal DAG and an optimal solution.

maximal and the following constraints are satisfied:

$$\forall j, \qquad \sum_i \max_k \left(\chi_k(\mathcal{N}_j,\mathcal{N}_i)\right) \leq d_j \qquad (1)$$

$$\forall j, \qquad \sum_k \sum_i \chi_k(\mathcal{N}_j,\mathcal{N}_i) w_k \leq b_j \qquad (2)$$

where $\chi_k(\mathcal{N}_j,\mathcal{N}_i) = 1$ if the edge $(\mathcal{N}_j,\mathcal{N}_i)$ belongs to T_k and 0 otherwise. Equations (1) and (2) model respectively the degree constraint and the capacity constraint at node $\mathcal{N}_j$.

We can also characterize the optimal broadcast rate using theorems from [12], [13], [14] that relate the optimal broadcast rate with the minimum source-cut of a weighted graph. Indeed, let us consider a weighted directed graph $G = (V, E, c)$, where $c(\mathcal{N}_j,\mathcal{N}_i)$ denotes the bandwidth allocated to the communication between $\mathcal{N}_j$ and $\mathcal{N}_i$. A j-cut of G is given by two sets of nodes S and S' such that $S \bigcup S' = V$, $\mathcal{N}_0 \in S$ and $\mathcal{N}_j \in S'$. The value of a cut (S, S') is the sum of the weights of all edges from a node in S to a node in S', and we denote by cut(j) the minimum value of a j-cut of G. This value cut(j) denotes the maximal value of a flow between the source node $\mathcal{N}_0$ and $\mathcal{N}_j$ and therefore represents an upper bound of the broadcast rate. Moreover, it is proven in [12] that this bound is actually tight, *i.e.* that the optimal broadcast rate using for graph G is equal to mincut$(G) = \min_j$ cut(j). At last, efficient algorithms [14] have been designed to compute the set of weighted trees that achieves this optimal broadcast rate.

Therefore, we can also formalize the problem of finding the optimal broadcast rate as follows:

BCASTFLOWS: Let b_j and d_j denote the outgoing bandwidth and outdegree of $\mathcal{N}_j$ (the source node being denoted as $\mathcal{N}_0$). Find an allocation of weights $c(\mathcal{N}_j,\mathcal{N}_i)$ such that mincut(G) is maximal and the following constraints are satisfied:

$$\forall j, \qquad |\{i, \quad c(\mathcal{N}_j,\mathcal{N}_i) > 0\}| \leq d_j \qquad (3)$$

$$\forall j, \qquad \sum_i c(\mathcal{N}_j,\mathcal{N}_i) \leq b_j \qquad (4)$$

Here also, equation (3) models the degree constraints and equation (4) models the capacity constraints.

III. Complexity Results

In this section, we prove that the decision problems **BCASTTREESDEC** and **BCASTFLOWSDEC** associated to **BCASTTREES** and **BCASTFLOWS** respectively are NP-Complete. In order to establish this result, we first prove (Section III-A) that **BCASTTREESDEC** and **BCASTFLOWSDEC** belong to NP, by showing how to restrict the search to compact solutions, where both the number of used trees, the weight of these trees and the flow on the edges have polynomial size in the size of the original instance. Then, we prove in Section III-B that finding the optimal set of weighted trees is NP-Complete in the strong sense.

A. Compact versions of Decision Problems

We look for versions **BCASTTREESCOMPACT** and **BCASTFLOWSCOMPACT** of **BCASTTREESDEC** and **BCASTFLOWSDEC** respectively, such that both problems are equivalent and such that we can restrict the search of a solution of **BCASTTREESCOMPACT** to set of weighted trees and flow values that can be encoded in polynomial size in the size of the original instance.

Indeed, let us consider the decision problems associated to **BCASTTREES** and **BCASTFLOWS**.

BCASTTREESDEC: Let b_j and d_j denote the outgoing bandwidth and outdegree of $\mathcal{N}_j$ (the master node is denoted as $\mathcal{N}_0$) and let $B > 0$. Find a weighted set of trees (w_k, T_k), with $w_k > 0$ such that constraints (1), (2) and $\sum_k w_k \geq B$ are satisfied.

BCASTFLOWSDEC: Let b_j and d_j denote the outgoing bandwidth and outdegree of $\mathcal{N}_j$ (the master node is denoted as $\mathcal{N}_0$) and let $B > 0$. Find an allocation of weights $c(\mathcal{N}_j,\mathcal{N}_i)$ such that such that constraints (3), (4) and mincut$(G) \geq B$ are satisfied.

It is not clear that **BCASTFLOWSDEC** belongs to NP since a solution may involve exponential size coefficients $c(\mathcal{N}_j,\mathcal{N}_i)$. Therefore, we need to consider compact versions

of **BCASTFLOWSDEC**. In order to bound the size of the coefficients, we rely on the linear programming approach to compute the optimal broadcast rate $\lambda*$, proposed in [15], [16], where a few constraints have been added in order to enforce degree constraints. In the following linear program, $(\mathcal{N}_j, \mathcal{N}_i)$ denotes the edge between $\mathcal{N}_j$ and $\mathcal{N}_i$ and special edges $(\mathcal{N}_j, \mathcal{N}_0)$ between $\mathcal{N}_j$ and $\mathcal{N}_0$ have been added to make the formulation more concise.

$$\text{Maximize } \lambda * \text{ subject to}$$
$$\begin{cases} \forall j, \quad \lambda* \leq f_j(\mathcal{N}_j, \mathcal{N}_0) \\ \forall j,k, \forall i \neq 0, \quad f_k(\mathcal{N}_j, \mathcal{N}_i) \leq c(\mathcal{N}_j, \mathcal{N}_i) \\ \forall j,k, \quad \sum_i (f_k(\mathcal{N}_j, \mathcal{N}_i) - f_k(\mathcal{N}_i, \mathcal{N}_j)) = 0 \\ \forall j,k, \quad \sum_i f_k(\mathcal{N}_j, \mathcal{N}_i) \leq b_j \\ \forall j,k, \quad \sum_i \delta_j^i \leq d_j \\ \forall i,j,k, \quad \delta_j^i \in \{0,1\} \text{ and } f_k(\mathcal{N}_j, \mathcal{N}_i) \leq \delta_j^i \\ \forall i,j,k \quad f_k(\mathcal{N}_j, \mathcal{N}_i), c(\mathcal{N}_j, \mathcal{N}_i) \text{ and } \lambda* \geq 0 \end{cases}$$

where $f_k(\mathcal{N}_j, \mathcal{N}_i)$ denotes the flow to $\mathcal{N}_k$ shipped on the edge $(\mathcal{N}_j, \mathcal{N}_i)$ and δ_j^i indicates if $(\mathcal{N}_j, \mathcal{N}_i)$ belongs to the overlay. The above mixed linear program provides the optimal rate $\lambda*$ among all broadcast solutions that satisfy both degree and bandwidth constraints. Once the optimal set of edges defined by the δ_j^i have been determined, the optimal rational values of the $f_k(\mathcal{N}_j, \mathcal{N}_i)$s and $c(\mathcal{N}_j, \mathcal{N}_i)$s can be found by solving a linear program in rational number, what can be done in polynomial time. Thus, it is possible to determine a bound B_1, that is polynomial in the size of the original instance (using Cramer formula for instance), such that the size of any coefficient $f_k(\mathcal{N}_j, \mathcal{N}_i)$ or $c(\mathcal{N}_j, \mathcal{N}_i)$ is smaller than B_1 in any optimal solution. Therefore, we can consider the following compact decision problem:

BCASTFLOWSCOMPACT: Let b_j and d_j denote the outgoing bandwidth and outdegree of $\mathcal{N}_j$ (the master node is denoted as $\mathcal{N}_0$) and let $B > 0$. Find an allocation of weights $c(\mathcal{N}_j, \mathcal{N}_i)$ such that the size of the $c(\mathcal{N}_j, \mathcal{N}_i)$s is smaller than B_1 and constraints (3), (4) and mincut$(G) \geq B$ are satisfied.

The linear programming formulation proves that **BCASTFLOWSCOMPACT** and **BCASTFLOWSDEC** are equivalent. Moreover, since $c(\mathcal{N}_j, \mathcal{N}_i)$ coefficients have polynomial size, we can check in polynomial time that a set of $c(\mathcal{N}_j, \mathcal{N}_i)$ values satisfies the degree constraints, the capacity constraints and the flow constraint. Therefore, we have proven the following theorem:

Theorem 3.1: **BCASTFLOWSCOMPACT** $\in$ NP.

It is not clear that **BCASTTREESDEC** belongs to NP since a solution may consist in an exponential number of trees with exponential size coefficients. Therefore, we need to consider compact versions of **BCASTTREESDEC**. In order to bound the possible number of trees and the size of the coefficients, we rely on the equivalence between the minimum source-cut and the broadcast rate. In order to determine the optimal set of weighted trees, we rely on the following theorem [17], vol. B, Chapter 53:

Theorem 3.2: Let $G = (V, E, c(\mathcal{N}_j, \mathcal{N}_i))$ be an oriented graph with weighted edges. There exist k_T weighted trees (λ_k, T_k) such that

$$\forall j,i, \quad \sum_k \lambda_k \chi_k(j,i) \leq c(\mathcal{N}_j, \mathcal{N}_i)$$

where $\chi_k(j,i) = 1$ if $(\mathcal{N}_j, \mathcal{N}_i) \in T_k$ and 0 otherwise, and $\sum_k \lambda_k = \lambda*$. Besides, these trees can be found in polynomial time, and $k_T \leq |V|^3 + |E|$.

Therefore, we can restrict the search of the optimal set of weighted trees to sets consisting of at most n^3+n^2 trees, and since the size of all coefficients $c(\mathcal{N}_j, \mathcal{N}_i)$ have polynomial size in the size of the instance, there exists a constant B_2, polynomial in the size of the original instance and such that $\forall k$, the size of λ_k is bounded by B_2. Thus, let us consider the following compact version of the decision problem:

BCASTTREESCOMPACT: Let b_j and d_j denote the outgoing bandwidth and outdegree of $\mathcal{N}_j$ (the master node is denoted as $\mathcal{N}_0$) and let $B > 0$. Find a weighted set of at most $n^3 + n^2$ trees (w_k, T_k), with $w_k > 0$, such that the size of the w_ks is smaller than B_2 and such that constraints (1), (2) and $\sum_k w_k \geq B$ are satisfied.

Since the number of trees and their weights have polynomial size, constraints (1) and (2) can be checked in polynomial time. Therefore, we have proved the following theorem:

Theorem 3.3: **BCASTTREESCOMPACT** $\in$ NP.

B. NP Completeness Results

In the previous section, we have proved that **BCASTTREESCOMPACT** and **BCASTFLOWSCOMPACT** are equivalent. Therefore, it is enough to prove that **BCASTFLOWSCOMPACT** is NP-Complete. In order to prove this result, we will make a reduction from the problem **3 PARTITION**.

3 PARTITION: Let $a_i, \quad 1 \leq i \leq 3p$ be $3p$ integers, such that $\forall j < 3p, \quad a_j \leq a_{j+1}$, $\sum_1^{3p} a_i = pT$ and $\forall i, \quad \frac{T}{4} < a_i < \frac{T}{2}$. Is there a partition of the a_is into p disjoint sets $S_i, \quad 1 \leq i \leq p$ containing exactly 3 elements and such that each set sums up to exactly T?

3 PARTITION is a well known NP-Complete problem in the strong sense [18]. Given a particular instance of **3 PARTITION**, let us consider the following instance $\mathcal{I}$ of **BCASTFLOWSCOMPACT**.

- There are $4p + 1$ nodes in three groups, and $B = T$,
- Group 1: $\forall 0 \leq j \leq p-1$, $b_j = 2T$ and $d_j = 4$,
- Group 2: $\forall p \leq j \leq 4p-1$, $b_j = T - a_{j-p+1}$ and $d_j = 1$,
- Group 3: $b_{4p} = 0$ and $d_{4p} = 0$.

Lemma 3.4: $\mathcal{I}$ has a solution if the instance of **3 PARTITION** has a solution.

Let us assume that the instance of **3 PARTITION** has a solution, *i.e.* there exists a partition of the a_is into p disjoint sets S_i, $1 \leq i \leq p$, containing exactly 3 elements and such that each set sums up to exactly T. Let us first organize the nodes as a chain, with the nodes of Group 1, Group 2 and Group 3 in this order. In this chain, nodes of Group 1 serve their successor at rate T and all the other nodes serve their successor with their whole bandwidth. Once the chain has been built, all the nodes in Group 1 have remaining bandwidth T and 3 remaining connections, and all the nodes in Group 2 and Group 3 have exhausted both their bandwidth and their degree. Similarly, all nodes in Group 1 and the first node of Group 2 have received the whole message, whereas the $(j+1)$-th node in Group 2 has received exactly $(T-a_j)$ and the node of Group 3 has received $(T-a_{3p})$. If a_j is in set S_i in the solution of **3 PARTITION**, we connect the node that receives $T-a_j$ to the i-th node of Group 1, with an edge of capacity a_j. Thus, all the nodes receive the complete message. Moreover, since each set S_i consists in 3 elements summing up to exactly T, the nodes in Group 1 can serve a whole set S_i with their 3 remaining connections and their remaining bandwidth T, what achieves the proof of the lemma.

Lemma 3.5: The instance of **3 PARTITION** has a solution if $\mathcal{I}$ has a solution.

Let us assume that $\mathcal{I}$ has a solution, *i.e.* there exists an overlay weighted graph, such that all degree and bandwidth constraints are satisfied, and all nodes receive a flow at least T.

Let us first estimate the overall available outgoing bandwidth. The overall available bandwidth from nodes of Group 1 is $p \times 2T$, the overall outgoing bandwidth from nodes of Group 2 is $\sum_1^{3p}(T-a_i) = 3pT - pT = 2pT$. Therefore, the overall outgoing bandwidth is $4pT$. On the other hand, there are $4pT$ nodes (all but the source) that need to receive the message at rate T, so that the overall required incoming bandwidth is at least $4pT$. Therefore, all the nodes use their full bandwidth.

In particular, node j from Group 2 communicates with exactly one neighbor at rate $T-a_{j-p+1}$ since its degree is one and no bandwidth can be wasted. Moreover, no node can receive a message from 2 nodes of Group 2, since $\forall i,j\ (T-a_i)+(T-a_j) > T$, so this would induce a bandwidth waste. Therefore, at least $3p$ nodes receive a message with rate $T-a_i$ for some i.

Let us now consider the number of incoming connections, and let us denote by n_1 the number of nodes that receive the whole message through one connection only (at rate T) and by n_2 the nodes that receive the message from at least 2 sources. Then, $n_1+n_2 = 4p$, the overall number of nodes that need to receive the message. Moreover, we have seen that at least $3p$ nodes receive a message with rate $T-a_i$ for some i and therefore $n_2 \geq 3p$, so that $n_1 \leq p$.

The overall incoming degree is smaller than the sum of the degrees of all nodes and larger than n_1+2n_2, so that $n_1+2n_2 \leq 7p$. Using $n_2 = 4p-n_1$, this implies that $n_1 \geq p$ and therefore $n_1 = p$ and $n_2 = 3p$. Moreover, these $3p$ nodes receive the message from exactly 2 sources (no more) and all $7p$ possible outgoing connections are actually used with positive bandwidth.

There are p nodes that receive the message from exactly one source. Since the nodes in Group 2 and 3 have bandwidth strictly smaller than T, these messages are sent from node in Group 1. Moreover, since all the nodes in Group 1 must make use of all their bandwidth ($2T$) and all their connections (4) with positive rate, a node in Group 1 cannot serve more than 1 node at rate T. Therefore, all nodes in Group 1 serve exactly 1 node at rate T and 3 other nodes with strictly smaller rates.

Therefore, p nodes receive the message from 1 source and $3p$ nodes from two sources with rates $T-a_i$ and a_i for some value i. Thus, all nodes from Group 1 need to send 3 a_i values to 3 different nodes such that a_i values sum up to T, what achieves the proof of the lemma.

We have therefore proved the following theorem:

Theorem 3.6: **BCASTFLOWSCOMPACT** is NP-Complete in the strong sense.

IV. Optimal Algorithm using Resource Augmentation

In this section, we propose an algorithm to build an overlay (interconnection) graph that achieves the optimal broadcast rate. This algorithm makes use of *resource augmentation*: it may increase the outdegree of $\mathcal{N}_j$ to at most $\max(d_j+2, 4)$.

A. Upper bound

We will first prove an upper bound on the achievable throughput for a given instance $\mathcal{I}$ of **BCASTFLOWS**. Let $\mathcal{S}$ be a solution that achieves throughput $T > 0$ on this instance. Since it is useless to communicate between two nodes at a rate larger than T, each $\mathcal{N}_i$ cannot use an outgoing bandwidth larger than $X_i = \min(b_i, Td_i)$. Since the overall sending rate must be equal to nT, we have

$$\sum_{i=0}^{n} \min(b_i, Td_i) \geq nT \tag{5}$$

The largest value T^* such that equation (5) is satisfied is thus an upper bound on the optimal throughput. It is interesting to note that T^* can be computed by dichotomic search. Indeed, the function $f(T) = \sum_{i=0}^{n} X_i - nT$ is continuous, piecewise affine, with decreasing slopes (each time a value Td_i becomes larger than b_i for some index i, the slope is decreased by d_i). Since $f(0) = 0$, f is increasing for small values of T ($\sum_i d_i$ has to be larger than n if there is a solution to $\mathcal{I}$) and f tends to $-\infty$ for large values of T, we know that there is only one value T^* such that

$\sum_{i=0}^{n} \min(b_i, T^* d_i) = nT^*$. Furthermore, another (trivial) upper bound on the achievable throughput is b_0, since the source has to be able to send the whole message at least once.

B. Algorithm Based on Resource Augmentation

We propose an algorithm that takes as input a value $T \leq b_0$ satisfying equation (5) and returns a solution that achieves throughput T, while requiring a slightly larger out-degree at some nodes. The first step of our algorithm consists in arranging all nodes (except the master node) by decreasing values of their outgoing capacity $X_i = \min(b_i, Td_i)$. We will thus assume in what follows that $X_1 \geq X_2 \geq \cdots \geq X_n$.

1) Easy case: A first and simple (but suboptimal) version of the algorithm works for any value T such that $\sum_{i=0}^{n-1} X_i \geq nT$. Note that since the X_is are sorted by decreasing values, then $\forall k < n, \sum_{i=0}^{k} X_i \geq (k+1)T$. In the following, we will denote $S_k = \sum_{i=0}^{k} X_i$.

The principle of this algorithm, formalized in Algorithm 1, is to satisfy (*i.e.* send a complete message to) the nodes in the previously defined sorting order, while maintaining the property that after each step, at most one node receives the message only partially (all the previous nodes receive the message at rate T and the following ones do not receive anything yet). A given $\mathcal{N}_k$ will thus send data to a consecutive set of nodes, say from $\mathcal{N}_{a_k}$ to $\mathcal{N}_{b_k}$. All intermediate nodes, except possibly a_k and b_k, will be be served at rate T. Since the total bandwidth used by $\mathcal{N}_k$ is X_k and $X_k \leq Td_k$, there are at most $d_k - 1$ such intermediate nodes. Hence the total out-degree of $\mathcal{N}_k$ is at most $d_k + 1$.

Furthermore, if we consider the directed graph where there is an edge between nodes i and j iff $\mathcal{N}_i$ serves $\mathcal{N}_j$ with positive rate, this algorithm produces an acyclic graph. Indeed, before each step k, the property $S_{k-1} \geq kT$ ensures that the bandwidth available so far (S_{k-1}) is enough to satisfy all nodes from 1 to k. Hence, each $\mathcal{N}_k$ will only serve nodes with strictly larger indexes (*i.e.* $a_k > k$ with the above notations).

Algorithm 1 Simple Algorithm

```
Set t = 1 and ∀j, r_j = T and ∀k, c_k = X_k
for k = 0 to n do
  while c_k > 0 do
    c(N_k, N_t) := min(r_t, c_k)
    c_k := c_k − c(N_k, N_t); r_t := r_t − c(N_k, N_t)
    if r_t = 0 then
      t := t + 1
    end if
  end while
end for
```

2) General Case: This section presents how to construct a solution that achieves the upper bound (5). We have seen in Section II that it may be necessary to introduce cycles in the topology in order to reach the optimum throughput. We propose here a procedure to build such cycles that reaches the optimum throughput while maintaining a low out-degree.

The first part of this algorithm consists in executing Algorithm 1 until the smallest index k_0 such that $S_{k_0} < (k_0+1)T$. The result is a partial solution in which all nodes up to k_0-1 are served at rate T, and nodes with indexes k_0 or larger do not send nor receive anything. In what follows, we will call such a solution a (k_0-1)-partial solution. In the next part, we build successive k-partial solutions for k equal to k_0, k_0+1, ..., until n.

For all values $k \geq k_0$, let $M_k = kT - S_{k-1}$ be the missing flow at node $\mathcal{N}_k$ when the bandwidth of all previous nodes are fully used. In a k-partial solution, it is compulsory that $\mathcal{N}_k$ sends a flow M_k for the total input and output flow rates to be equal. Let $R_k = X_k - M_k$ be the remaining capacity at $\mathcal{N}_k$ in such a solution. These definitions imply the following property: $R_k + M_{k+1} = X_k - kT + S_{k-1} + (k+1)T - S_k = T$. The following proof is a (constructive) proof by induction that it is possible to build successive k-partial solutions such that:

- $c(\mathcal{N}_k, \mathcal{N}_{k-1}) + c(\mathcal{N}_{k-1}, \mathcal{N}_k) = T$,
- the out-degree of $\mathcal{N}_k$ is at most 2,
- the out-degree of $\mathcal{N}_{k-1}$ is at most 3,
- and remaining available bandwidth of $\mathcal{N}_k$ is R_k.

For simplicity of the presentation, we first assume $n > k_0$ (the particular and simpler case $n = k_0$ will be discussed later).

Initial case, $k = k_0$: We start from the $(k-1)$-partial solution built using Algorithm 1. In this solution, $\mathcal{N}_k$ already receives a flow $T - M_k$ from a set of nodes $\mathcal{A}$ (that all receive the message at rate T). We select an arbitrary edge $(\mathcal{N}_u, \mathcal{N}_v)$ with capacity at least M_k[1]. Since $n \geq k+1$, we set $\alpha = \frac{M_{k+1}}{T}(T - M_k)$ and $\beta = \frac{M_{k+1}}{T} M_k$ and make the following modifications (depicted in Figure 2):

- Flow α goes from $\mathcal{A}$ to $\mathcal{N}_{k+1}$ instead of $\mathcal{N}_k$;
- Flow M_k goes from u to $\mathcal{N}_k$ instead of $\mathcal{N}_v$;
- $\mathcal{N}_k$ sends a flow $R_k + \beta$ to $\mathcal{N}_{k+1}$;
- $\mathcal{N}_k$ sends a flow $M_k - \beta$ to $\mathcal{N}_v$;
- $\mathcal{N}_{k+1}$ sends a flow β to $\mathcal{N}_v$ and a flow α to $\mathcal{N}_k$.

The choice of α and β ensures that the flow on all edges remains positive, that no node exceeds its outgoing bandwidth, and also that $\alpha + \beta = M_{k+1}$ and therefore $R_k + \beta + \alpha = T$. Hence, all the nodes receive a flow at rate T from the source node. Therefore, we have built a (k_0+1)-partial solution, with $c(\mathcal{N}_{k_0+1}, \mathcal{N}_{k_0}) + c(\mathcal{N}_{k_0}, \mathcal{N}_{k_0+1}) = T$, and the outdegrees of nodes k_0 and k_0+1 are both equal to 2. Finally, $\mathcal{N}_{k+1}$ sends a flow $\alpha + \beta$, and thus has $X_{k+1} - M_{k+1} = R_{k+1}$ remaining available bandwidth.

[1]The fact that $T \leq b_0$ ensures that $\mathcal{N}_1$ necessarily receives T from the source node, hence such an edge always exists.

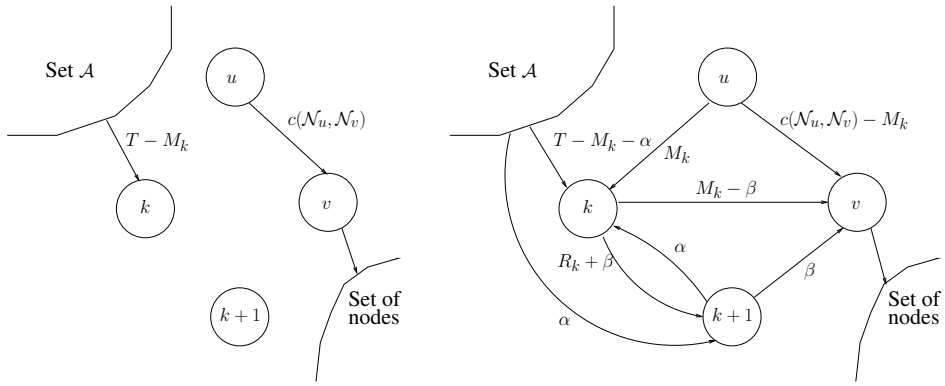


Figure 2. Modifications for the initial case.

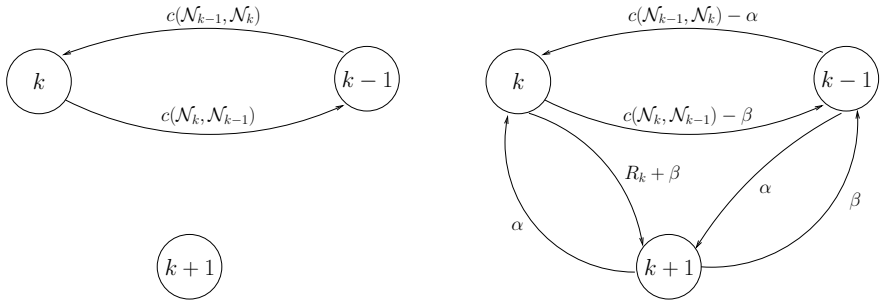


Figure 3. Modifications when adding $\mathcal{N}_k + 1$.

Induction: Let us assume that we have built a k-partial solution satisfying above properties. $\mathcal{N}_k + 1$ is inserted as follows (see Figure 3):

- $\mathcal{N}_k$ uses all its remaining bandwidth R_k to send data to $\mathcal{N}_{k+1}$;
- Part α of the flow going from $\mathcal{N}_{k-1}$ to $\mathcal{N}_k$ now goes through $\mathcal{N}_{k+1}$;
- Part β of the flow going from $\mathcal{N}_k$ to $\mathcal{N}_{k-1}$ now goes through $\mathcal{N}_{k+1}$.

Once again, we set $\alpha = \frac{M_{k+1}}{T} c(\mathcal{N}_{k-1}, \mathcal{N}_k)$ and $\beta = \frac{M_{k+1}}{T} c(\mathcal{N}_k, \mathcal{N}_{k-1})$, thus ensuring $\alpha + \beta = M_{k+1}$. It is easy to check that nodes k and $k-1$ receive a flow T from the source node (in the same way as in the original k-partial solution, with just a diversion through $\mathcal{N}_{k+1}$). Furthermore, $\mathcal{N}_{k+1}$ receives a flow α from $\mathcal{N}_{k-1}$, a flow $c(\mathcal{N}_{k-1}, \mathcal{N}_k) - \alpha$ from $\mathcal{N}_{k-1}$ through $\mathcal{N}_k$, and a flow $c(\mathcal{N}_k, \mathcal{N}_{k-1})$ from node k. Note that these three flows are compatible, since the total flow going from $\mathcal{N}_k$ to $\mathcal{N}_{k+1}$ is $c(\mathcal{N}_{k-1}, \mathcal{N}_k) - \alpha + c(\mathcal{N}_k, \mathcal{N}_{k-1}) = T - \alpha = T - M_{k+1} + \beta = R_k + \beta$, *i.e.* the capacity of the edge.

This solution is thus a $(k+1)$-partial solution, with $\mathcal{N}_{k+1}$ having out-degree 2 and $\mathcal{N}_k$ having out-degree at most 3 (one edge has been added to the previous k-partial solution). This concludes the proof.

If $n = k_0$: In that case, induction is not necessary. The algorithm simply applies the transformation described in the initialization phase, with $\alpha = \beta = 0$ and the remaining bandwidth R_{k_0} is ignored.

Overall solution: In the solution obtained by recursively applying above procedure, the actual out-degree o_i of $\mathcal{N}_i$ is at most $\max(d_i + 2, 4)$. Indeed,

- in the $(k_0 - 1)$-partial solution obtained at the end of algorithm 1, $o_i \leq d_i + 1$;
- during the initialization phase, the outdegree of $\mathcal{N}_u$ and of one node of the set $\mathcal{A}$ increases by 1;
- when adding $\mathcal{N}_{k+1}$, the outdegree of node $\mathcal{N}_{k-1}$ is increased by 1, and it was at most 3.

V. Simulations

In this section, we analyze the performance of our algorithms through simulations on random instances, by comparing them to simpler solutions, as well as to an unconstrained upper bound.

A. Algorithms

Unconstrained solution: In order to analyze the effect of the degree constraint on the achievable throughput, we compute the best achievable throughput without the degree constraint. The easiest way to do this is simply to set all

$d_i = n$ and then run our algorithm. The resulting throughput is then $\frac{\sum_{i=0}^{n} b_i}{n}$.

Best Tree: Building trees is a simple and common way of performing broadcast operations. We thus analyze the maximum throughput that can be achieved using a single tree. In a tree-based solution that achieves throughput T, all nodes receive data from one parent only, so that all edges must ship flow T. If T is known, the maximal number e_i of outgoing edges node $\mathcal{N}_i$ can have is given by $e_i = \min(d_i, \lfloor \frac{b_i}{T} \rfloor)$. Hence, since a tree with $n+1$ nodes (the master node being included) has n edges, we must have $\sum_i e_i \geq n$. Reciprocally, any value of T such that this property holds can be achieved with a tree solution in which node i has e_i sons. Therefore, the maximum throughput that can be achieved using a single tree is the maximal value of T such that $\sum_i \min(d_i, \lfloor \frac{b_i}{T} \rfloor) \geq n$.

Best Acyclic: The best throughput of an acyclic solution is obtained by applying Algorithm 1. Indeed, in an acyclic solution one node cannot use its outgoing bandwidth, and hence such a solution of throughput T must satisfy $\sum_{i=0}^{n-1} X_i \geq nT$.

Cyclic with Resource Augmentation: This corresponds to the algorithm described in Section IV-B2.

Note that both BESTACYCLIC and CYCLICRA make use of resource augmentation. Indeed, each node may have an out-degree d_i+1 for BESTACYCLIC, and $\max(d_i+2, 4)$ for CYCLICRA. In order to make a fair comparison between all algorithms, the degrees of the nodes are increased before computing the BESTTREE and BESTACYCLIC throughput, so that the degrees actually used are the same for all algorithms.

B. Random Instance Generation

We generate instances randomly, trying to focus on realistic scenarios. To this end, we have used traces of node incoming and outgoing bandwidths from the XtremLab[2] project, which is part of the BOINC distributed computing platform. This project was used to monitor all nodes of this platform, and report information about their computing and communication capabilities. We have used the upload bandwidth values, previously filtered to the interval $1,000 - 100,000$ in order to remove the outstanding values that come from people cheating by lying about their capacities. A heterogeneity ratio of 100 is the order of magnitude than can be expected in a distributed platform. Furthermore the results obtained in this section do not depend on the actual interval used for filtering. The trace represents more than $7,500$ values. The complementary cumulative density function is shown on Figure 4. Random instances are obtained by selecting b_is uniformly at random from these values. In our simulations, all nodes have a constant maximal degree $d_i = d$, the value of d depending on the considered scenario.

[2]http://xw01.lri.fr:4320/

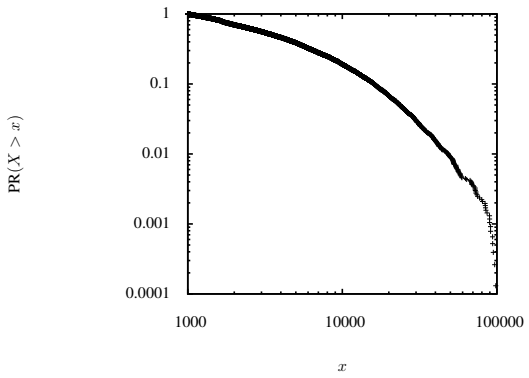


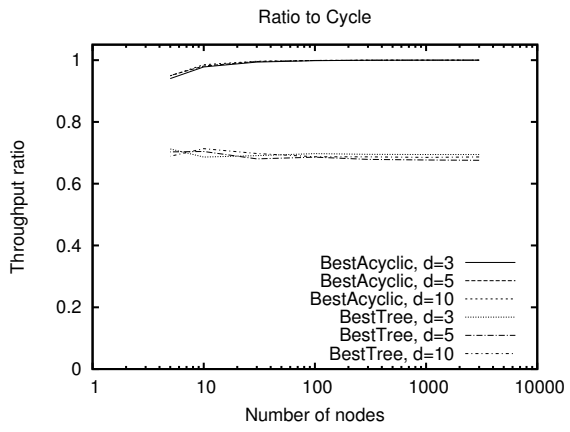
Figure 4. Complementary CDF of the distribution used to generate bandwidth values.

C. Simulation results

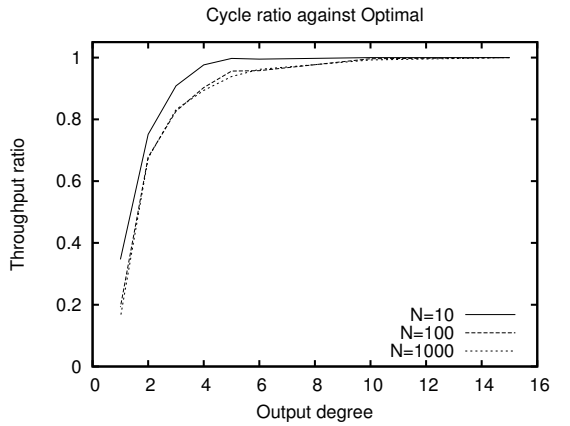
In a first set of experiments, we have compared the performance of the three algorithms: BESTTREE, BESTACYCLIC, and CYCLICRA, for varying values of n (the number of nodes) and d (the maximum allowed degree). For a given instance, we compute the ratio of the throughput of the first two algorithms over the throughput obtained by CYCLICRA. We generate 30 instances for each (n, d) pair, and plot the average ratio over these instances. The result of these experiments is shown on Figure 5(a) (x axis is logscale).

We can see that the results do not depend on the values of n and d. The throughput of BESTTREE is significantly lower (about 30%) than the one reached by the two other algorithms. However, the throughputs of BESTACYCLIC and CYCLICRA are almost indistinguishable, except for low values of n. In this case, the fact that BESTACYCLIC does not use the bandwidth of the last node degrades the throughput more significantly.

In a second set of experiments, we have compared the performance of CYCLICRA to UNCONSTRAINED, the best achievable throughput without degree constraint. The comparison has been done for varying values of n and d, and the results are depicted in Figure 5(b). Of course, the ratio is quite low for small values of d, since the nodes with large outgoing bandwidth cannot make good use of it. However, we can see that the throughput is within 10% of the upper bound as soon as n reaches 7, and is within 1% for n larger than 10. The fact that the necessary degree to obtain quasi-optimal throughput is low is of great practical importance, since the maintenance cost of an overlay network is directly related to the number of neighbors a node must handle. Another interesting remark is that in all simulations, the value of k_0 for CYCLICRA was either $n-1$ or $n-2$, which means that only the very last nodes need to set up cycles to reach the optimum throughput, while the largest part of the overlay is actually acyclic.



(a) Comparison of algorithms to CYCLICRA



(b) Comparison of CYCLICRA to UNCONSTRAINED

Figure 5. Simulation results.

VI. Related Works and Implementation Issues

Broadcasting on heterogeneous platforms has been the focus of a vast literature, especially since overlay networks have become an effective alternative to IP multicast. In this paper, we consider at the same time the design of the overlay network (with bounded number of neighbors) and the broadcast operation itself. The design of efficient overlay networks, based of a set of trees or a mesh, is addressed in [6], [19], whereas the design of efficient broadcast algorithm in heterogeneous networks has been considered in [20], [21], [22].

In [7], Hong and Prasanna consider the problem of allocating a large number of heterogeneous tasks under the bounded multi-port model. The communication model they consider is slightly different since they assume that the topology of the underlying physical network is known and that contentions take place on network links rather than at the network interfaces of the nodes. They prove that maximizing the throughput (*i.e.* the number of tasks processed by the platform once steady state has been reached) can be expressed as a flow problem, whose solution can be computed in a distributed way using [23]. The independent tasks scheduling problem under the bounded degree multi-port model has been addressed in [24].

In [25], the authors study a problem similar to ours, slightly less constrained, which they call Peer Assisted Streaming Capacity. In their setting, the degrees of all nodes are equal, excpet for the source which has unbounded degree. Furthermore, they use the tree formulation of the problem, and consider per-tree degree bounds: a given node may only have d sons in a given tree, but they may differ from one tree to another. The same authors consider global per-node degree bounds in [26], and provide a $\frac{1}{2}$-approximation algorithm, as well as a randomized distributed scheme.

In [16], Massoulié et al. propose several distributed algorithms to optimize the throughput of a broadcast. They consider several models. In the first one, the topology of the network is known and a bandwidth is associated to each edge. In this context, they propose a very simple distributed algorithm: each processor compares the set of packets it has received to the sets of packets received by its neighbor nodes, and sends a new packet to the neighbor node that has received the less packets. Remarkably enough, they prove that this fully distributed algorithm, based on local control only, achieves a throughput arbitrarily close to the optimal. The proof uses a sophisticated analysis on randomized algorithms. They also consider a model closely related to ours, where each node is characterized by its outgoing bandwidth, but the topology is a complete graph and they do not consider degree constraints.

In Section IV, we have shown how to compute the optimal throughput under a more sophisticated model, where the maximal number of simultaneous connections a node can handle is bounded. Moreover, the proposed algorithm also computes (i) an overlay network in which the degree constraints are satisfied and (ii) the flow associated to each edge, *i.e.* the size of data that should be sent on each edge. This information is nevertheless not enough for practical implementation, since we need to know what should be sent. To obtain a practical implementation, several solutions exist. First, the proof provided for the existence of the flow in Section IV-B can be adapted to associate a set of intervals of $[0, 1]$ to each edge so that each node exactly receives the interval $[0, 1]$ from its incoming neighbors. Second, we can use the results of [14] (see Section II) to compute the weighted set of trees associated to a given flow. The third solution consists in applying the algorithm of Massoulié et al. [16] to the overlay graph where edge capacities are flow values. With this last solution, we obtain a fully distributed algorithm that achieves a throughput arbitrarily close to the optimal solution. Moreover, Massoulié's algorithm requires for a node to exchange the set of received packets with its neighbors. Thus, bounding the degree of each node eases

the practical implementation of this algorithm and limits its control cost. At last, the simulations we made on realistic instances from BOINC traces prove that in practice, a small degree is enough to achieve close to optimal throughput.

VII. Conclusions

We have considered the problem of broadcasting a large message onto an heterogeneous platform. We introduced the bounded degree multi-port model to model the communication capabilities of the nodes. This model encompasses the benefits of both bounded multi-port model and one-port model and its parameters can be easily instantiated at runtime. Based on this model, we formalized the broadcast problem and proved that it is NP-Complete in the strong sense. Nevertheless, we have provided a polynomial time algorithm that achieves the optimal broadcast rate at the price of a small resource augmentation of the degree of the nodes. The performance of this algorithm against several basic heuristics and against the optimal broadcast rate (without degree limitation) has been assessed through simulations, based on realistic parameter sets observed in volunteer computing platforms. Simulations prove that the algorithm performs surprisingly well on such instances, in the sense that it is possible to achieve optimal broadcast rate using a small degree at each node. This property is of great importance since the cost for maintaining the overlay in distributed networks is directly related to the size of the neighborhood. At last, we have shown that the output of our algorithm provides enough information to use fully decentralized and randomized algorithms that are known to achieve optimal broadcast rate.

References

[1] S. Johnsson and C. Ho, "Optimum broadcasting and personalized communication in hypercubes," *IEEE Transactions on Computers*, vol. 38, no. 9, pp. 1249–1268, 1989.

[2] J. Watts and R. Geijn, "A pipelined broadcast for multidimensional meshes," *Parallel Processing Letters*, vol. 5, no. 2, pp. 281–292, 1995.

[3] Y. Tseng, S. Wang, and C. Ho, "Efficient broadcasting in wormhole-routed multicomputers: anetwork-partitioning approach," *IEEE Transactions on Parallel and Distributed systems*, vol. 10, no. 1, pp. 44–61, 1999.

[4] X. Zhang, J. Liu, B. Li, and Y. Yum, "CoolStreaming/DONet: A data-driven overlay network for peer-to-peer live media streaming," in *Proceedings IEEE INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, 2005.

[5] L. Vu, I. Gupta, J. Liang, and K. Nahrstedt, "Mapping the PPLive network: Studying the impacts of media streaming on P2P overlays," *Department of Computer Science, University of Illinois at Urbana-Champaign, Tech. Rep. UIUCDCS*, pp. 2006–275.

[6] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: high-bandwidth multicast in cooperative environments," *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5, pp. 298–313, 2003.

[7] B. Hong and V. Prasanna, "Distributed adaptive task allocation in heterogeneous computing environments to maximize throughput," *International Parallel and Distributed Processing Symposium*, 2004.

[8] O. Beaumont, A. Legrand, L. Marchal, and Y. Robert, "Pipelining Broadcasts on Heterogeneous Platforms," *IEEE Transactions on Parallel and Distributed Systems*, pp. 300–313, 2005.

[9] T. Saif and M. Parashar, "Understanding the Behavior and Performance of Non-blocking Communications in MPI," *Lecture Notes in Computer Science*, pp. 173–182, 2004.

[10] M. A. Brown, "Traffic Control HOWTO. Chapter 6. Classless Queuing Disciplines," *http://tldp.org/HOWTO/Traffic-Control-HOWTO/classless-qdiscs.html*, 2006.

[11] B. Hubert *et al.*, "Linux Advanced Routing & Traffic Control. Chapter 9. Queueing Disciplines for Bandwidth Management," *http://lartc.org/lartc.pdf*, 2002.

[12] J. Edmonds, "Edge disjoint branchings," in *Combinatorial Algorithms: Courant Computer Science Symposium 9: January 24-25, 1972*. Algorithmics Press, 1972, p. 91.

[13] L. Lovasz, "On two minimax theorems in graph," *Journal of Combinatorial Theory (B)*, vol. 21, pp. 96–103, 1976.

[14] H. Gabow and K. Manu, "Packing algorithms for arborescences (and spanning trees) in capacitated graphs," *Mathematical Programming*, vol. 82, no. 1, pp. 83–109, 1998.

[15] Z. Li, B. Li, D. Jiang, and L. Lau, "On achieving optimal throughput with network coding," in *Proceedings IEEE INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, 2005.

[16] L. Massoulie, A. Twigg, C. Gkantsidis, and P. Rodriguez, "Randomized decentralized broadcasting algorithms," in *IEEE INFOCOM 2007. 26th IEEE International Conference on Computer Communications*, 2007, pp. 1073–1081.

[17] A. Schrijver, *Combinatorial optimization: polyhedra and efficiency*. Springer, 2003.

[18] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*. WH Freeman San Francisco, 1979.

[19] D. Kostić, A. Rodriguez, J. Albrecht, and A. Vahdat, "Bullet: high bandwidth data dissemination using an overlay mesh," in *SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles*. New York, NY, USA: ACM, 2003, pp. 282–297.

[20] P. Liu and T. Sheng, "Broadcast scheduling optimization for heterogeneous cluster systems," in *Proceedings SPAA'00*. ACM New York, NY, USA, 2000, pp. 129–136.

[21] A. Bar-Noy and C. Ho, "Broadcasting multiple messages in the multiport model," *IEEE Transactions on Parallel and Distributed Systems*, vol. 10, no. 5, pp. 500–508, 1999.

[22] S. Khuller and Y. Kim, "Broadcasting in heterogeneous networks," *Algorithmica*, vol. 48, no. 1, pp. 1–21, 2007.

[23] A. Goldberg and R. Tarjan, "A new approach to the maximum-flow problem," *Journal of the Association for Computing Machinery. Vol*, vol. 35, no. 4, pp. 921–940, 1988.

[24] O. Beaumont, L. Eyraud Dubois, H. Rejeb, and C. Thraves, "Allocation of Clients to Multiple Servers on Large Scale Heterogeneous Platforms," in *IEEE (ICPADS'09)*, Shenzhen Chine, 2009. [Online]. Available: http://hal.archives-ouvertes.fr/inria-00413437/en/

[25] S. Liu, R. Zhang-Shen, W. Jiang, J. Rexford, and M. Chiang, "Performance bounds for peer-assisted live streaming," in *SIGMETRICS '08: Proceedings of the 2008 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*. New York, NY, USA: ACM, 2008, pp. 313–324.

[26] S. Liu, M. Chen, M. Chiang, S. Sengupta, J. Li, , and P. A. Chou, "P2p streaming capacity under node degree bound," Tech. Rep., 2009. [Online]. Available: http://www.princeton.edu/~chiangm/infocom10long.pdf