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

Olivier Beaumont, Nicolas Bonichon, Lionel Eyraud-Dubois, Przemyslaw Uznanski, Shailesh Kumar Agrawal

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

Olivier Beaumont, Nicolas Bonichon, Lionel Eyraud-Dubois and Przemysław Uznański
INRIA Bordeaux – Sud-Ouest, University of Bordeaux, LaBRI , France
Shailesh Kumar Agrawal - Citibank, Singapore

***Abstract*—We consider the classical problem of broadcasting a large message at an optimal rate in a large scale distributed network under the multi-port communication model. In this context, we are interested in both building an overlay network and providing an explicit algorithm for scheduling the communications. From an optimization point of view, we aim both at maximizing the throughput (*i.e.* the rate at which nodes receive the message) and minimizing the degree of the participating nodes, *i.e.* the number of TCP connections they must handle simultaneously. The main novelties of our approach are the introduction of this degree constraint and the classification of the set of participating nodes into two parts: open nodes that stay in the open-Internet and "guarded" nodes that lie behind firewalls or NATs. Two guarded nodes cannot communicate directly, but rather need to use an open node as a gateway for transmitting a message. In the case without guarded nodes, we prove that it is possible to reach the optimal throughput, at the price of a quasi-optimal (up to a small additive increase) degree of the participating nodes. In presence of guarded nodes, our main contributions are a closed form formula for the optimal cyclic throughput and the proof that the optimal solution may require arbitrarily large degrees. In the acyclic case, we propose an algorithm that reaches the optimal throughput with low degree. Then, we prove a worst case ratio between the optimal acyclic and cyclic throughput and show through simulations that this ratio is on average very close to 1, what makes acyclic solutions efficient both in terms of throughput maximization and degree minimization.**

## I. Introduction

Data dissemination in distributed platforms has been the subject of a vast literature. The problem comes into two flavors, depending on the context. On the one hand, if the topology of the platform is known (in the case 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 a large scale Internet level platforms, the goal is to find the topology (*i.e.* the overlay network) that maximizes the throughput.

The one-to-all broadcast, or single-node broadcast, is the most primary collective communication pattern: initially, only the source processor holds 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], [4]).

The same framework applies for broadcasting a live stream of data, such as a movie or a TV show. In the context of content distribution systems, it is at the core of live streaming distribution systems such as CoolStreaming [5], PPLive [6] or SplitStream [7]. In this case also, we are interested in the distribution of a large message to all the nodes of a large scale platform, made of a large number of computers, geographically distributed, and interconnected by the Internet. In the context of this work, it is thus not possible to obtain the actual topology of the core of the network, and we are rather interested in application-level solutions. Thus, the goal is to build an overlay network 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 the context of large scale Internet platforms, it is common to assume that the communication between two nodes is only limited by the available outgoing bandwidth of the sender and by the incoming bandwidth of the receiver. This assumption is also very suited to the case where nodes are connected to the Internet with low bandwidth links, like DSL for example. In that case, the bandwidth limitation is either physical (from the link capacity) or logically enforced at the user's request. In large scale platforms, it is also desirable to limit the number of connections that can be handled simultaneously at each node. Both of these assumptions

are common in the context of data dissemination in large scale platforms. However, they fail to correctly model the behavior of the nodes located behind a NAT or a firewall. As we will see, adding this constraint on node connectivity capabilities strongly modifies the algorithms and the theoretical results.

In summary, our goal is first 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, such that nodes behind a NAT or a firewall use third party nodes to communicate and such that the overall throughput that can be reached using this overlay network is close to the optimal one. One of the major contributions of this paper is to study, under a realistic communication model and for a classic communication scheme, the impact on the complexity and on the performance of the algorithms of having nodes lying behind NATs and firewalls.

This paper is organized as follows. In Section II, we review some of the relevant literature to further explain the positioning of our work, then we introduce the notations and models that we will use throughout the paper. In Section III, we present complexity results for our problem, and we study a simple case to familiarize the reader with the problem and the techniques used in the paper. This allows us to present a summary of the results of this paper at the end of Section III. In Section IV, we analyze *acyclic* solutions and provide algorithms that build optimal and low degree solutions. In Section V, we solve the problem in the absence of firewalls by extending the simple algorithm presented in Section III. Section VI provides worst-case and average case comparisons between the throughput achievable by cyclic and acyclic solutions. Concluding remarks are given in Section VII.

Some of the results of this paper were presented in two different conference papers [8], [9].

## II. Models and Paper Positioning

### A. Platform Modeling

In the context of large scale distributed platforms where Internet is the underlying network, it is not realistic to assume that the topology is known, especially in a dynamic context. Nevertheless, several embedding tools have been proposed, whose goal is to map a set of nodes onto a metric space [10], [11] (*i.e.* to give them coordinates) so that their distance in the metric space is a good approximation of the metric of interest (usually the latency between two nodes or the bandwidth that a point-to-point communication between them can achieve). In the case of latencies, a very well-known embedding tool is Vivaldi [12], which embeds nodes into a 2D+1 metric space and relies on direct measurements to dynamically adapt node coordinates. For bandwidth estimation, a good candidate is DMF [13], which builds a summary of the distance matrix thanks to low-rank matrix factorization. Furthermore, it has been recently proven experimentally [14] on the PlanetLab dataset that a good estimation accuracy can be obtained with the classical last mile assumption (or bounded multiport model), in which each node is associated to an incoming and an outgoing bandwidth limit, and where the achievable bandwidth between $C_i$ and $C_j$ is the minimum of the outgoing bandwidth of $C_i$ and the incoming bandwidth of $C_j$.

The bounded multiport model has already been advocated by Hong et al. [15] for independent tasks distribution on heterogeneous platforms. In this model, node $C_i$ can communicate with any number of nodes $C_j$ simultaneously, each using a bandwidth $c_{i,j}$, provided that its outgoing bandwidth is not exceeded, *i.e.*, $\sum_j c_{i,j} \leq b_i^{\text{out}}$. Similarly, node $C_i$ can receive messages from any number of nodes $C_j$ simultaneously, each using a bandwidth $c_{j,i}$, provided that its incoming bandwidth is not exceeded, *i.e.*, $\sum_j c_{j,i} \leq b_i^{\text{in}}$. 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. But in the context of large scale platforms, in which the networking heterogeneity ratio may be high, it is unreasonable to assume that a 10GB/s server may be kept busy for 10 seconds while communicating a 10MB data file to a 1MB/s DSL node. Therefore, in our context, we will assume that all communications are directly handled at TCP level. Nevertheless, in order to keep the flavor of the one-port model, we will minimize the number of connections that need to be handled simultaneously at a given node. This constraint is particularly important in a context where QoS mechanisms are used to fix or bound the bandwidth associated to each communication (each TCP connection in practice). It is worth noting that at the operating system level, several QoS mechanisms enable a prescribed sharing of bandwidth [16], [17], [18]. In particular, it is possible to handle simultaneously several connections and to fix the bandwidth allocated to each connection. In our context, it has been proved in [19] that these mechanisms are necessary since the bandwidth allocated to the connection between $C_i$ and $C_j$ may be lower than both $b_i^{\text{out}}$ and $b_j^{\text{in}}$. Therefore, the variant of the

LastMile 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, which is compulsory in the context of large scale distributed platforms. Practical implementation is achieved by using TCP QoS mechanisms and by bounding the number of connections.

However, this model fails to correctly model the behavior of the nodes located behind a NAT or a firewall. This issue is crucial in the context of Peer-to-Peer applications running over the Internet. For instance, in distributed applications such as Skype [20], [21] or Bittorent [22], NATs play a crucial role, since in certain situations where "hole punching" techniques [23] fail, it can be impossible for a pair of nodes to communicate directly. In this case, the technique consists in using a third party node that acts as a relay for the packets. At a higher level, we can classify the nodes between open and guarded nodes, where open-open, open-guarded (and guarded-open) connections are possible, but not guarded-guarded. As we will see, adding this constraint on node connectivity capabilities strongly modifies the algorithms and the theoretical results.

### B. Related works

Broadcast and streaming optimization have already been the subject of several studies in the literature. However, none of them has considered the constraint added by the presence of firewalls in the system. The work closest to our approach is by Liu et al [24] in which they provide bounds for the streaming rate, the upload rate of the source needed to ensure a given stream rate, and the depth of the distribution trees produced. Degree constraints are also considered in their work, but with specific limitations. In particular, the degree of the source is not limited, and the degree constraint on nodes is considered separately for each tree of the solution, which means that the actual degree of each node is not limited.

More applied studies have been published, which focus on designing distributed algorithms to build the streaming overlay. For example, CoolStreaming [5] builds upon a gossip-based overlay to propose a distributed streaming algorithm. This algorithm inherently includes degree limitations, and provides a guarantee on the diameter of the overlay, but no guarantee about the streaming rate is available. On the other hand, SplitStream [7] is based on a distributed hash table and builds an overlay made of $k$ different distribution trees, with a probabilistic guarantee on the streaming rate. Furthermore, SplitStream allows multicast (some nodes may only receive the data from a subset of the $k$ trees – but they do not choose which part) and also includes a degree limitation, which is typically $k$ times larger than the degree of our solutions.

### C. Positioning

In this paper, we assume that the network can be represented using the LastMile model. To obtain this representation, we rely on tools such as Bedibe (see https://gforge.inria.fr/projects/bedibe/ and [14]), that extract from a reasonable size of point-to-point measurements the values of the parameters of the LastMile model (degrees and bandwidths) in a time that is compatible with the dynamics of the network.

The contribution of this paper consists in computing, using this instantiated model, the overlay network (which nodes communicate together) and the bandwidths that should be allocated to each edge of the overlay in order to maximize the overall throughput of the collective communication scheme, given the bandwidth, the degree and the connectivity constraints of the network. The resulting weighted graph can be decomposed into a set of weighted broadcast trees [25, vol B, Chapter 53]. This decomposition specifies which data should be sent on which edge at a given time step.

In order to avoid this decomposition step, which is difficult to use in practice, we rely on the randomized broadcasting algorithm proposed by Massoulié in [4]. This algorithm is fully decentralized and is even able to deal with small variations of resource performance due to its randomized and dynamic nature. This algorithm requires knowledge of the topology of the network with bandwidths on edges and no contentions on the nodes, which is in general not realistic. On the other hand, the overlay network that we build has exactly, by construction, these properties, provided that bandwidth sharing mechanisms are used for the communications in order to limit the bandwidth of a communication to the weight of the edge, such as proposed in [16], [17], [18]. The overlay network that we build in this paper can therefore be used as direct inputs of Massoulié's algorithm.

Therefore, by relying on the one hand on Bedibe to instantiate the parameters of the LastMile model and on the other hand on Massoulié's algorithm to actually perform the broadcast operation, our algorithmic contribution provides a practical solution to the streaming problem whose approximations (due to the model, to the use of approximation algorithms for NP-Complete problems and to the decentralized and randomized implementation of the broadcast) can be rigorously analyzed and

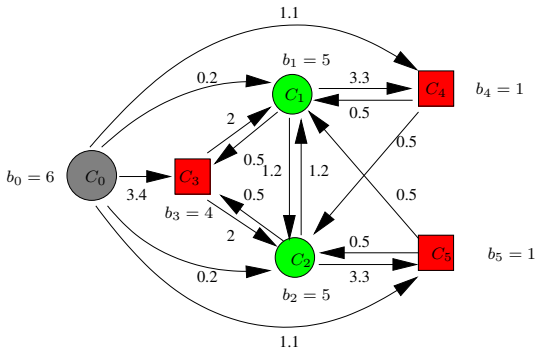


Figure 1. An instance with $n = 2$ open nodes (plus the source), $m = 3$ guarded nodes together with an optimal broadcast scheme of throughput 4.4. In this broadcast scheme, the outdegree of the source is $o_0 = 5$, the outdegree of every guarded node is $o_3 = o_4 = o_5 = 2$ and the outdegree of the two open nodes is $o_1 = o_2 = 3$.

controlled. In this paper, we focus on the approximation algorithms perspective.

### D. Model and notations

We consider a situation in which a *source* node, denoted as $C_0$, wants to broadcast a message. The recipients are partitioned into two sets: on the one hand some nodes belong to the *open-Internet*, and can communicate with each other freely (we call them *open* nodes); on the other hand some nodes can communicate only with nodes of the open-Internet, because they are behind a firewall or behind a NAT router (we call them *guarded* nodes). The source itself is supposed to be an open node.

An instance of our problem is specified by the number $n$ and $m$ of open and guarded nodes, and by the outgoing bandwidth $b_i$ of each node $C_i$ for $i \in [\![0, n+m]\!]$. The source node is $C_0$, nodes $C_i$ for $i \in \mathcal{O} = [\![1, n]\!]$ are open nodes, and nodes $C_i$ for $i \in \mathcal{G} = [\![n+1, n+m]\!]$ are guarded nodes.

The output of the problem is a broadcast scheme, defined by values $\{c_{i,j} | (i,j) \in [\![0, n+m]\!]^2\}$, where $c_{i,j}$ indicates the rate at which node $C_i$ sends data to node $C_j$, subject to the following constraints:

- $\forall i \in [\![0, n+m]\!], \sum_j c_{i,j} \leq b_i$ (bandwidth constraint)
- $\forall (i,j) \in \mathcal{G}^2, c_{i,j} = 0$ (firewall constraint).

We implicitly assume that the input bandwidth of each participating node is large enough. The *throughput* of a broadcast scheme is given by $T = \min_{i \in [\![1, n+m]\!]} \{maxflow(C_0 \to C_i)\}$, where the flows are computed on the weighted graph described by the $c_{i,j}$s. Furthermore, given a broadcast scheme, we can define the *outdegree* of a node $C_i$ as the number of nodes to which $C_i$ actually sends some data, *i.e.* $o_i = |\{j,\ c_{i,j} > 0\}|$. Notations are illustrated in Figure 1.

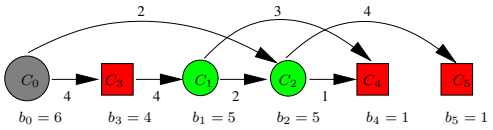


Figure 2. An acyclic broadcast scheme of throughput 4 on the instance of Figure 1. The order associated with this acyclic broadcast scheme is $\sigma = 031245$.

As stated above, we want to provide broadcast schemes with small degree. To define what small degree may mean, let us note that in a solution of throughput $T$, the weight of any edge $c_{i,j}$ is at most $T$ (indeed, receiving data at a rate larger than $T$ is useless). Hence, if node $i$ uses all of its outgoing bandwidth, then its outdegree $o_i$ is at least $\lceil \frac{b_i}{T} \rceil$ and small degree therefore means $o_i$ close to $\lceil \frac{b_i}{T} \rceil$. A solution that achieves (a fraction of) $T^*$ by using an outdegree $o_i \leq \lceil \frac{b_i}{T^*} \rceil + d$ is thus a $d$-additive resource augmentation (approximation) algorithm. As a consequence, we do not consider strict degree constraints, but rather analyze the outdegrees used by our solutions in terms of $\lceil \frac{b_i}{T} \rceil$.

Computing a solution that achieves throughput $T^*$ and such that the degree of each node is at most $\lceil \frac{b_i}{T^*} \rceil$ turns out to be a NP-complete problem (see Section III-A), even for the special case where all nodes are open ($m = 0$). We are thus interested in this paper in *approximate* solutions, both in terms of throughput (with respect to $T^*$) and additive resource augmentation on the degrees (with respect to $\lceil \frac{b_i}{T^*} \rceil$).

We prove that the situation differs if we concentrate on *acyclic* or more general *cyclic* solutions. A broadcast scheme is said to be *acyclic* if its communication graph (represented by the matrix $c$) is acyclic, which is equivalent to the existence of an order $\sigma$ on the nodes such that

$$\forall i, j \in [\![0, n+m]\!], i > j \Rightarrow c_{\sigma(i),\sigma(j)} = 0.$$

This condition states that $\sigma(i)$, the node at position $i$ in the ordering $\sigma$ cannot feed $\sigma(j)$, the node at position $j$, if $i > j$. Figure 2 shows an example of an acyclic broadcast scheme associated with the order $\sigma = 031245$.

For a given instance and a given order $\sigma$, we denote by $T^*_{ac}(\sigma)$ the optimal acyclic scheme compatible with the order $\sigma$. For a given instance, we denote by $T^*_{ac}$ the optimal acyclic throughput:

$$T^*_{ac} = \max_{\sigma} T^*_{ac}(\sigma)$$

## III. Simple Case and Summary of Results

As we have already noted, the problem comes in several flavors. Among the parameters that strongly influence the complexity of the problem are (i) the presence of guarded nodes (nodes behind firewalls) and (ii) the acyclicity of the solution, what leads to four different problems. In Section III-A, we prove that all four problems are NP-Complete. In Section III-B, we concentrate on the easiest case, *i.e.* the case where we are looking for an acyclic solution with open nodes only. At last, in Section III-C, we summarize the results proved in later sections for all the four different problems, together with the bounds on their relative performance.

### A. Complexity Results

*1) Introduction:*

In the case where guarded nodes are present, some of the communications are forbidden, what makes the combinatorial structure of the problem more complex. On the other hand, searching for an acyclic solution limits the search space and we will see throughout this paper that it actually makes the problem easier. Of course, cyclic solutions can achieve higher throughput than acyclic ones, but we will see later in Section VI, that this only holds up to a (small) ratio $5/7$.

*2) NP-Completeness:*

*Theorem 3.1:* Finding an optimal allocation while satisfying the degree constraints (keeping $o_i \leq \left\lceil \frac{b_i}{T} \right\rceil$) is NP-Complete in the strong sense.

We prove in Appendix VIII this result by reduction to the **3 Partition** problem. Note that this NP-Completeness result applies to all four different situations. Indeed, let us first remark that considered instances only contain open nodes, so that the NP-Completeness a fortiori holds in (the more complicated case of) presence of guarded nodes. Second, we a priori search for a general cyclic solution but we nevertheless prove that the optimal throughput (for our instances) can be reached using an acyclic solution (see Figure 8), so that above proof also shows that finding the optimal acyclic solution is NP-Complete.

### B. Acyclic Solution with open nodes only

*1) Introduction:* We consider the simplest case, where all nodes are open and we search for an acyclic solution. As we have just proved it, despite its apparent simplicity, this problem is NP-Complete in the strong sense. Nevertheless, we will prove that it is possible to achieve the optimal throughput at the price of a very small additive increase of the degree of the nodes. To establish this result, we will first prove an upper bound on the achievable throughput of any acyclic solution. Then, we will exhibit an algorithm that achieves this throughput while keeping the degree of the nodes small. This algorithm will be used as a starting point to build a cyclic solution for instances without guarded nodes in Section V, and it introduces some of the ideas that will be later adapted in Section IV for instances with guarded nodes.

*2) Upper Bound:* The first idea which will be used throughout the paper is that nodes should be ordered by non-increasing order of bandwidth. In the remainder, we will thus consider that nodes are ordered so that $b_1 \geq \ldots \geq b_n$ and we will denote $S_k = \sum_{i=0}^{k} b_i$. In any acyclic solution, nodes can be sorted in topological order such that a node only feeds nodes with larger indexes. In particular, there exists at least one node that does not send data to any other node. Therefore, the overall throughput achieved by any acyclic solution $T^*$ is upper bounded by $\frac{S_{n-1}}{n}$ since $b_n$ denotes the smallest capacity and $n$ nodes $C_1, \ldots, C_n$ must receive the message at a rate $T^*$. Additionally, it is clear that $T^* \leq b_0$.

*3) Algorithm:* Let us now describe an algorithm that provides an optimal acyclic solution for instances without guarded nodes.

The algorithm takes as input $T^* = \min(b_0, \frac{S_{n-1}}{n})$, and returns a broadcast scheme that achieves throughput $T^*$. Let us first remark that since the $b_i$s are sorted in non-increasing order, and since $T^* \leq b_0$, then $\forall 0 \leq k < n, S_k \geq (k+1)T^*$.

The basic principle of the algorithm formalized in Algorithm 1 is to satisfy (*i.e.* send a complete message to) the nodes one after the other (considered in the previously defined sorting order), while maintaining the property that after each step, at most one node receives the message only partially, *i.e.* all previous nodes receive the message at rate $T^*$ and all following ones do not receive anything yet. $C_i$ thus sends data to a consecutive set of nodes, say from $C_{\alpha_i}$ to $C_{\beta_i}$. All intermediate nodes, except possibly $\alpha_i$ and $\beta_i$, will be served at rate $T^*$. Since the total bandwidth used by $C_i$ is $b_i$, there are at most $\left\lceil \frac{b_i}{T^*} \right\rceil - 1$ such intermediate nodes. Hence the number of nodes served at least partially by $C_i$, *i.e.* its outdegree, is at most $\left\lceil \frac{b_i}{T^*} \right\rceil + 1$.

The behavior of Algorithm 1 is depicted in Figure 3.

Furthermore, this algorithm produces an acyclic graph. Indeed, before each step $i$, the property $S_{i-1} \geq iT^*$ ensures that the bandwidth available so far ($S_{i-1}$) is always large enough to satisfy all nodes from 1 to $i$. Hence, each $C_i$ will only serve nodes with strictly larger indexes (*i.e.* $\alpha_i > i$ with above notations).

From above remarks, we can conclude that Algo-

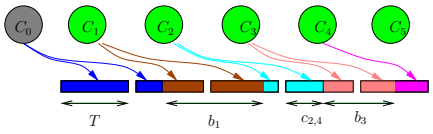


Figure 3. Solution returned by Algorithm 1. The upper part represents how the capacity of $C_i$ is used (in column $i$) and the lower part describes which nodes the data sent to $C_i$ (in column $i$) come from.

**Algorithm 1** Acyclic Algorithm on open nodes only.

```
Set t = 1 and ∀i, r_i = T* and ∀i, s_i = b_i
for i = 0 to n do
    while s_i > 0 do
        c_{i,t} := min(r_t, s_i)
        s_i := s_i − c_{i,t}; r_t := r_t − c_{i,t}
        if r_t = 0 then
            t := t + 1
        end if
    end while
end for
```

rithm 1 builds an acyclic communication graph such that each node receives exactly $T^*$ from nodes with smaller indexes and that the degree of $C_i$ is at most $\left\lceil \frac{b_i}{T^*} \right\rceil + 1$. Therefore, Algorithm 1 returns a solution of optimal throughput in which nodes have a degree that is larger by at most an addictive factor of 1 with respect to the lower bound. Therefore, unless $P \neq NP$, no algorithm can always return in polynomial time a better solution (in terms of minimal increase in the degree constraint)

### C. Summary of the results

Now that the reader is familiar with the problem we consider, we can state the results proved in this paper.

1) In the presence of **open and guarded nodes** or with **open nodes only**, finding the best **acyclic** solution or finding the best **cyclic** solution is NP-Complete in the strong sense (See Section III-A).
2) In the presence of **open nodes only**, the optimal **acyclic** throughput can be achieved at the price of a small linear increase of **1** in the degree of the nodes (see Section III-B).
3) In the presence of **open and guarded nodes**, the optimal **acyclic** throughput can be achieved at the price of a small linear increase of **3** in the degree of the nodes (see Section IV).
4) In the presence of **open nodes only**, the optimal **cyclic** throughput can be achieved at the price of a small linear increase of **2** in the degree of the nodes (see Section V).
5) In the presence of **open and guarded nodes**, the optimal **cyclic** throughput can be achieved only with an **unbounded increase** in the degree of the nodes (see Section V).
6) For any instance, the optimal **acyclic** throughput is at least **5/7** of the optimal **cyclic** throughput (see Section VI).
7) On average (see Section VI), for a wide variety of realistic scenarios, the throughput of the **low degree acyclic solutions** proposed by our algorithms are very close to the **optimal cyclic** throughput (at most 5% decrease).

Therefore, except in the cyclic case with open and guarded nodes, and despite the strong NP-Completeness result, it is possible to build low degree solutions that achieve optimal throughput at the price of a small increase in the degree bound. Moreover, if the complexity of proofs dramatically increases from the acyclic open case to the cyclic open case and to the acyclic guarded case, all proposed algorithms are very efficient in time complexity and can therefore be used in practice.

The situation strongly differs in the cyclic guarded case since arbitrarily large degrees are required in order to achieve optimal throughput, so that we cannot rely on small degree increases to obtain optimal performance. Nevertheless, we prove in Section VI that the algorithm that returns low degree solutions in the acyclic guarded case is a $\frac{5}{7}$-approximation algorithm for the cyclic guarded case.

## IV. Acyclic algorithm with guarded nodes

In this section, we describe how to build an acyclic broadcast scheme with a small increase in the degree constraint in presence of guarded nodes:

*Theorem 4.1:* Given an instance $I$ and a throughput $T$, it is possible to decide in linear time if $T \leq T^*_{ac}$. Moreover if $T \leq T^*_{ac}$, it is possible to compute in linear time a broadcast scheme of throughput $T$ such that

- for every guarded node $j \in \mathcal{G}$, outdegree $o_j$ is bounded: $o_j \leq \left\lceil \frac{b_j}{T} \right\rceil + 1$;
- for at most one open node $i$, $o_i \leq \left\lceil \frac{b_i}{T} \right\rceil + 3$;
- for all other open nodes, $o_i \leq \left\lceil \frac{b_i}{T} \right\rceil + 2$.

For instances with guarded nodes, there is no closed formula for $T^*_{ac}$, but the algorithm of Theorem 4.1 can be combined with a dichotomic search (on $T$) to find the optimal acyclic throughput.

The proof of Theorem 4.1 can be decomposed into three steps: we start by proving dominance relations in order to characterize optimal acyclic schemes (Lemma 4.3). We then provide an algorithm for testing

if throughput $T$ is achievable. If this is the case, a valid ordering is computed (Lemma 4.5). Then, we show how to compute a low degree solution from the computed valid ordering (Lemma 4.6).

### A. Dominance relations

Before entering into the details of the algorithm, let us start with an intuitive property of ordering of nodes. An ordering $\sigma$ is said to be *increasing* if its restriction to $\mathcal{O}$ is the identity on $\mathcal{O}$, and its restriction to $\mathcal{G}$ is the identity on $\mathcal{G}$. This means that nodes of the same color are ordered by non-increasing order on their bandwidth. The order $\sigma = 031245$ is an increasing order for the instance of Figure 2 whereas $\sigma = 041235$ is not increasing.

In Section III, we have proved (in the open nodes only case) that good solutions can be built with increasing orderings, and the next lemma (whose proof can be found in the appendix) shows that this also holds in the general case.

*Lemma 4.2 (proof in Appendix IX-A):*

$$T^*_{ac} = \max_{\sigma:\text{increasing}} \{T^*_{ac}(\sigma)\}.$$

An increasing order can be naturally encoded by a binary word $\pi$ with $n$ letters ◯ (corresponding to open nodes) and $m$ letters □ (corresponding to guarded nodes): it is sufficient to specify if $\sigma(i)$ belongs to $\mathcal{O}$ or to $\mathcal{G}$. We denote by $|\pi|$ the length of the word $\pi$, and by $|\pi|_{\bigcirc}$ (resp. $|\pi|_{\square}$) the number of letters ◯ (resp. □) in $\pi$. For instance, the word $\pi = \square\bigcirc\bigcirc\square\square$ encodes the increasing order $\sigma = 031245$ for the instance of Figure 2.

The notation $\pi' \sqsubseteq \pi$ (resp. $\pi' \sqsubset \pi$) means that $\pi'$ is a prefix (resp. a strict prefix) of $\pi$.

From now on, when no confusion is possible, $\pi$ will be identified with its corresponding increasing order. For instance, $T^*_{ac}(\pi)$ corresponds to the optimal acyclic throughput associated with the order encoded by $\pi$. A word $\pi$ is said to be *valid* (with respect to an instance $I$ and a throughput $T$) if $T^*_{ac}(\pi) \geq T$.

A solution $c$ is said to be *conservative* with respect to order $\sigma$, if there are no triplets of distinct indices $i, j, k$, such that $i < k$ and $j < k$, $\sigma(i) \in \mathcal{G}$, $\sigma(j), \sigma(k) \in \mathcal{O}$, and $c_{\sigma(j),\sigma(k)} > 0$ and $\sum_{l=i+1}^{k} c_{\sigma(i),\sigma(l)} < b_{\sigma(i)}$ simultaneously. The idea behind this definition is to consider solutions that feed the open nodes from guarded nodes as soon as possible. Indeed, the firewall constraint prevents transfer from guarded nodes to guarded nodes: transfer from open nodes is thus a valuable resource, and it is a "waste" to use it to feed open nodes when it is not necessary. Figure 2 shows an example of a conservative acyclic broadcast scheme and Figure 4 shows an example of a non-conservative one.

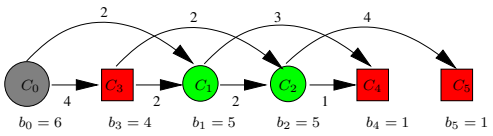


Figure 4. A non-conservative acyclic broadcast scheme: if we take $i = 1, j = 0, k = 2$, we see that node $C_1 = C_{\sigma(k)}$ could be totally fed by guarded node $C_3 = C_{\sigma(i)}$, but it uses the open bandwidth of the source $C_0 = C_{\sigma(j)}$.

This means that when creating a conservative solution incrementally (by satisfying the nodes in a given order $\sigma$), there is no choice for the type of nodes that should feed the next node to add: a guarded node must be fed by open nodes (because of the firewall constraint), and an open node should be fed by a guarded nodes as long as some of them have remaining outgoing capacity.

*Lemma 4.3 (proof in Appendix IX-B):* For every order $\sigma$ there exists a *conservative* solution $c$ that achieves $T^*_{ac}(\sigma)$.

Given a throughput $T$, and a coding word $\pi$ with $0 \leq i \leq n$ letters ◯ and $0 \leq j \leq m$ letters □, let $\mathcal{C}_\pi$ be the set of partial conservative solutions on the partial increasing order encoded by $\pi$ (that feeds nodes $C_1, \ldots, C_i$ and $C_{n+1}, \ldots, C_{n+j}$).

All partial conservative solutions of $\mathcal{C}_\pi$ have the same amount of available throughput of each type. Let us denote by $O(\pi)$ (respectively $G(\pi)$) the open (respectively guarded) bandwidth available at the end of the partial solutions of $\mathcal{C}_\pi$.

*Lemma 4.4 (proof in Appendix IX-C):*

$$G(\pi) = b_{n+1} + \ldots + b_{n+j} - i \cdot T + W(\pi) \quad (1)$$
$$O(\pi) = b_0 + b_1 + \ldots + b_i - j \cdot T - W(\pi) \quad (2)$$

and $O(\pi) + G(\pi) = \sum_{k=0}^{|\pi|_{\bigcirc}} b_k + \sum_{k=n+1}^{n+|\pi|_{\square}} b_k - |\pi|T$.

### B. Greedy algorithm

In this section, we present Algorithm 2 that decides whether a given throughput $T$ is feasible. If $T$ is feasible, Algorithm 2 also outputs a valid coding word. It works by iteratively building a partial conservative solution $\pi$, deciding at each step how to extend the partial solution (by ◯ or by □). This decision is made greedily, by choosing □ if it is possible. The algorithm is forced to take ◯ (see line 12):

- when it is not possible to choose □ at the current step ($O(\pi) < T$);
- or when choosing □ would make it impossible to continue afterwards ($O(\pi\square) + G(\pi\square) < T$).

Of course, if all guarded nodes have been used (line 6), the algorithm chooses ◯. Another special case is when

| $\pi$ | $\epsilon$ | $\square$ | $\square\bigcirc$ | $\square\bigcirc\square$ | $\square\bigcirc\square\bigcirc$ | $\square\bigcirc\square\bigcirc\square$ |
|---|---|---|---|---|---|---|
| $O(\pi)$ | 6 | 2 | 7 | 3 | 5 | 1 |
| $G(\pi)$ | 0 | 4 | 0 | 1 | 0 | 1 |
| $W(\pi)$ | 0 | 0 | 0 | 0 | 3 | 3 |

Table I
EXECUTION OF ALGORITHM 2 ON THE INSTANCE OF FIGURE 1. Observe that the amount of open-open transfer ($W(\pi)$) is only 3 whereas in the acyclic scheme proposed in Figure 2 this amount is 4.

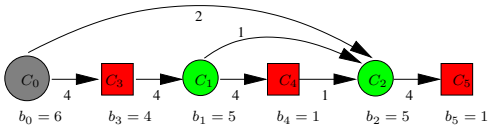


Figure 5. The acyclic broadcast scheme of throughput 4 built by Algorithm 2. The order associated with this scheme is $\sigma = 031425$.

only one guarded node is left. In that case (see lines 8-11), the algorithm chooses at each step the node with the largest $b_i$ (unless it is guarded and $O(\pi) < T$).

Table IV-B shows an execution of Algorithm 2 on the instance of Figure 1. The generated scheme is shown in Figure 5.

**Algorithm 2** GreedyTest ($T$)

```
1:  π ← ϵ
2:  while |π| < n + m do
3:      if O(π) + G(π) < T then return FAIL
4:      i ← |π|_◯ ; j ← |π|_□ ; l ← □
5:      if i ≠ n then
6:          if j = m then
7:              l ← ◯
8:          else if j = m − 1 then
9:              if O(π) < T or b_{n+j+1} < b_{i+1} then
10:                 l ← ◯
11:             end if
12:         else if O(π) < T or O(π□) + G(π□) < T
    then
13:             l ← ◯
14:         end if
15:     end if
16:     π ← πl
17:     if O(π) < 0 then return FAIL
18: end while
19: return π
```

The following lemma states that this algorithm is valid.

*Lemma 4.5 (proof in Appendix IX-D):* Given an instance $I$ and a throughput $T$, Algorithm 2 returns a valid word (a word $\pi$ such that $T^*_{ac}(\pi) \geq T$) if and only if $T$ is feasible for this instance ($T^*_{ac} \geq T$).

### C. Low degree scheme for a word $\pi$

The output of Algorithm 2 is an encoding word and an ordering, together with the amounts of guarded or open bandwidths used for this purpose, but not the actual values of the $c_{i,j}$s. There are several possibilities for the $c_{i,j}$s. However, in order to prove bounds on the degree of the nodes, we will feed each node by the earliest possible nodes with unused upload bandwidth (as in the open nodes case described in Section III-B).

*Lemma 4.6 (proof in Appendix IX-E):* From the word $\pi$ given by Algorithm 2, it is possible to build a broadcast scheme such that

- for every guarded node $j \in \mathcal{G}$, outdegree $o_j$ is bounded: $o_j \leq \left\lceil \frac{b_j}{T} \right\rceil + 1$;
- for at most one open node $i$, $o_i \leq \left\lceil \frac{b_i}{T} \right\rceil + 3$;
- for the other open nodes, $o_i \leq \left\lceil \frac{b_i}{T} \right\rceil + 2$.

Lemma 4.6 concludes the proof of Theorem 4.1.

## V. CYCLIC CASE

This section considers cyclic broadcast schemes. We start by giving an upper bound on the optimal cyclic throughput $T^*$:

*Lemma 5.1:*

$$T^* \leq \min\left(b_0, \frac{b_0 + O}{m}, \frac{b_0 + O + G}{n+m}\right),$$

where $O = \sum_{i=1}^{n} b_i$ and $G = \sum_{i=n+1}^{n+m} b_i$.
For the instance of Figure 1, $O = 10$, $G = 6$. Hence, from this lemma, we know that the throughput of the broadcast scheme of Figure 1 is optimal since $\min(6, 16/3, 22/5) = 4.4$.

*Proof:* Clearly $T^* \leq b_0$, since the whole message has to be sent at least once by the source. Then, the $m$ guarded nodes have to receive the message at rate $T^*$ and therefore consume $mT^*$ bandwidth. Since this bandwidth must come from the source and the open nodes, then $mT^* \leq b_0 + O$. Finally, all $n+m$ nodes must receive the whole message at rate $T^*$ and the bandwidth must come from the source, the open and the guarded nodes, so that $(m+n)T^* \leq b_0 + O + G$. ■

As shown in Figure 6, it is not always possible to achieve a solution of optimal throughput with low degree in presence of guarded nodes. Since we are interested in low degree solutions, the remainder of this section is restricted to the no guarded nodes case. For this special case, we present an algorithm with the following properties:

*Theorem 5.2 (proof in Appendix X-A):* There exists a polynomial time algorithm which takes as input any instance without guarded nodes and a target value of

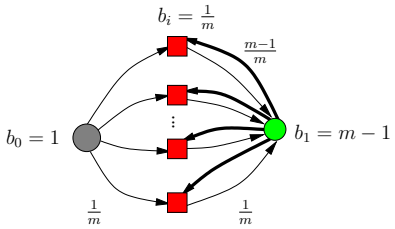


Figure 6. An instance with guarded and open nodes where the optimal cyclic throughput is $T^* = \min(b_0, \frac{b_0+b_1}{m}, \frac{b_0+b_1+mb_2}{m+1}) = 1$. In the optimal solution, the source has degree $m$, whereas $\left\lceil \frac{b_0}{T^*} \right\rceil = 1$.

$T \leq T^* = \min\left(b_0, \frac{b_0+O}{n}\right)$, and which builds a cyclic solution of throughput $T$, in which any node has outdegree $o_i \leq \max\left(\left\lceil \frac{b_i}{T} \right\rceil + 2, 4\right)$.

## VI. Cyclic/Acyclic throughput comparison

In this section, we compare the optimal acyclic throughput with the optimal (cyclic) throughput. On the one hand we show that the ratio $\frac{T^*_{ac}}{T^*}$ can be as small as $\frac{5}{7}$ for (small-size) instances and as small as $\frac{1+\sqrt{41}}{8}$ for arbitrary large instances (by contrast, when there are only open nodes, this ratio tends to one when the number of nodes is large). On the other hand, we show that this ratio is larger than $\frac{5}{7}$ for any instance, so that this bound is tight. Finally we present experimental results on the ratio $\frac{T^*_{ac}}{T^*}$ on random instances, that prove that acyclic solutions achieve much better results than the $\frac{5}{7}$ bound in practice. Due to space limitations, all proofs can be found in Appendix XI

### A. Worst cases

*1) Without guarded nodes:*

*Theorem 6.1 (proof in AppendixXI-A):* For any instance $I$ of size $n$ and without guarded nodes,

$$\frac{T^*_{ac}}{T^*} \geq 1 - \frac{1}{n}.$$

*2) With guarded nodes:* Our first result states that the optimal acyclic (low degree) solutions achieve a throughput that is at least $\frac{5}{7}$ of the optimal cyclic solution (with possibly arbitrarily large degree) and that this $\frac{5}{7}$ bound is tight.

*Theorem 6.2 (proof in AppendixXI-B):* For any instance, $\frac{T^*_{ac}}{T^*} \geq \frac{5}{7}$. Moreover, there exists an instance such that ratio is reached.

The second result states that the optimal acyclic throughput does not get arbitrarily close to the optimal cyclic throughput when the size of the instances grows.

*Theorem 6.3:* For every $\epsilon > 0$ and every $K \in \mathbb{N}$, there exist instances with at least $K$ open nodes and $K$ guarded nodes such that

$$\frac{T^*_{ac}}{T^*} \leq \frac{1+\sqrt{41}}{8} + \epsilon \approx 0.925 + \epsilon.$$

To conclude this subsection, we show an exhaustive exploration of all possible tight and homogeneous instances, for $n$ and $m$ between 0 and 100. For each of them we compute the ratio $T^*_{ac}/T^*$ (see Figure 7).

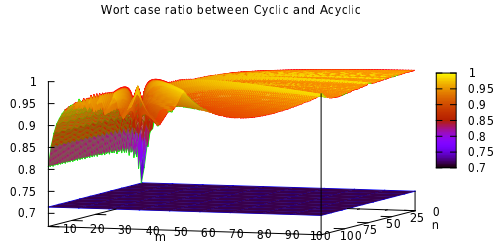


Figure 7. Worst case ratio between cyclic and acyclic optimal solutions on tight homogeneous instances. The bottom plane is $\frac{5}{7} \simeq 0.714$.

On the one hand, we can observe the result of Theorem 6.3: when $m \simeq \frac{\sqrt{41}-3}{8}n$ (for example $n = 100$ and $m = 42$), the ratio remains below 1, even for large values of $n$ and $m$. On the other hand we can observe that except for few small instances, the ratio $T^*_{ac}/T^*$ is larger than 0.8.

### B. Average cases

In addition to this worst-case analysis, we also analyze the average ratio between acyclic and cyclic throughput of randomly generated instances. In order to explore the performance of our algorithms in different heterogeneity conditions, we consider several probability distributions for the bandwidths of the nodes. Due to lack of space, the results and plots are given in Appendix XII.

## VII. Conclusion

We have considered the classical problem of broadcasting a large message at an optimal rate in a large scale distributed and heterogeneous network. We have advocated the use of the bounded degree multiport model, that encompasses the advantages of both the bounded multiport model and the 1-port model. The main originalities of our work is that we consider the case where some broadcast nodes lie in the open Internet whereas other broadcast nodes are guarded (behind NATs or firewalls), and that we search for either cyclic

or acyclic solutions. For three of the four different problems (namely acyclic/open acyclic/open and guarded and cyclic/open), we establish the complexity and provide algorithms with low degree (optimal up to a small constant additive term) that achieve optimal throughput. For the last problem (cyclic/open and guarded), reaching the optimal throughput may require arbitrarily large degree at some nodes, but on the other hand, we prove a tight worst-case bound of 5/7 for the ratio between acyclic and cyclic cases.

In summary, the main conclusions of our study are that guarded nodes can be taken into account with a low increase in complexity, and that acyclic solutions are sufficient to achieve high throughput. Furthermore, we believe that using simple yet theoretically solid methods (the algorithms from Massoulié and the bounded multiport last mile model) allows to derive theoretical results which can indeed be achieved in practice. In particular, our solution should be resilient to small variations in the communication performance of nodes. However it is probably not resilient to churn.

This work also opens many theoretical perspectives. Since the use of the bounded multiport model enables to design (quasi-)optimal solutions with respect to both degree and throughput, we can introduce new objectives, such as dealing with the churn of the platform or optimizing the depth of produced schemes in order to minimize delays.

## ACKNOWLEDGMENTS

This work is partially supported by ANR (Agence Nationale de Recherche), project reference ANR 08 SEGI 022 (USS SimGrid) and ANR 11 INFRA 13 (SONGS).

## 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: a network-partitioning approach," *IEEE Transactions on Parallel and Distributed systems*, vol. 10, no. 1, pp. 44–61, 1999.

[4] L. Massoulie, A. Twigg, C. Gkantsidis, and P. Rodriguez, "Randomized decentralized broadcasting algorithms," in *IEEE INFOCOM 2007*, 2007, pp. 1073–1081.

[5] 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*, vol. 3, 2005, pp. 2102–2111.

[6] L. Vu, I. Gupta, J. Liang, and K. Nahrstedt, "Mapping the PPLive network: Studying the impacts of media streaming on P2P overlays," University of Illinois at Urbana-Champaign, Tech. Rep. UIUCDCS-R-2006-2758, 2006.

[7] 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.

[8] O. Beaumont, L. Eyraud-Dubois, and S. Kumar, Agrawal, "Broadcasting on Large Scale Heterogeneous Platforms under the Bounded Multi-Port Model," in *24th IEEE International Parallel and Distributed Processing Symposium (IPDPS 2010)*, Atlanta, United States, 2010.

[9] O. Beaumont, N. Bonichon, L. Eyraud-Dubois, and P. Uznanski, "Broadcasting on Large Scale Heterogeneous Platforms with connectivity artifacts under the Bounded Multi-Port Model," in *ICPADS 2011 - 17th Internation Conference on Parallel and Distributed Systems*, Taïnan, Taiwan, 2011, pp. 173–180.

[10] T. Ng and H. Zhang, "Predicting internet network distance with coordinates-based approaches," in *IEEE INFOCOM 2002*, New York, NY, USA, 2002, pp. 170–179.

[11] J. Ledlie, P. Gardner, and M. Seltzer, "Network coordinates in the wild," in *4th USENIX Symposium on Networked Systems Design & Implementation*, 2007, pp. 299–311.

[12] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: a decentralized network coordinate system," in *SIGCOMM '04*. New York, NY, USA: ACM, 2004, pp. 15–26.

[13] Y. Liao, P. Geurts, and G. Leduc, "Network distance prediction based on decentralized matrix factorization," in *NETWORKING 2010*, M. Crovella, L. Feeney, D. Rubenstein, and S. Raghavan, Eds. Springer Berlin Heidelberg, 2010, vol. 6091, pp. 15–26.

[14] O. Beaumont, L. Eyraud-Dubois, and Y. Won, "Using the last-mile model as a distributed scheme for available bandwidth prediction," in *Proceedings of the EuroPar 2011*, 2011.

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

[16] A. B. Downey, "Tcp self-clocking and bandwidth sharing," *Computer Networks*, vol. 51, no. 13, pp. 3844 – 3863, 2007.

[17] D. Abendroth, H. van den Berg, and M. Mandjes, "A versatile model for tcp bandwidth sharing in networks with heterogeneous users," *AEU - International Journal of Electronics and Communications*, vol. 60, no. 4, pp. 267 – 278, 2006.

[18] Y. Zhu, A. Velayutham, O. Oladeji, and R. Sivakumar, "Enhancing tcp for networks with guaranteed bandwidth services," *Computer Networks*, vol. 51, no. 10, pp. 2788 – 2804, 2007.

[19] O. Beaumont and H. Rejeb, "On the importance of bandwidth control mechanisms for scheduling on large scale heterogeneous platforms," in *Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium on*, 2010, pp. 1–12.

[20] S. Baset and H. Schulzrinne, "An analysis of the skype peer-to-peer internet telephony protocol," *Arxiv preprint cs/0412017*, 2004.

[21] S. Guha, N. Daswani, and R. Jain, "An experimental study of the skype peer-to-peer VoIP system," in *Proceedings of IPTPS*, vol. 6, 2006.

[22] R. Jimenez, F. Osmani, and B. Knutsson, "Connectivity properties of mainline bittorrent DHT nodes," in *Peer-to-Peer Computing, 2009. P2P'09. IEEE Ninth International Conference on*, 2009, pp. 262–270.

[23] P. Srisuresh, B. Ford, and D. Kegel, "State of peer-to-peer (P2P) communication across network address translators (NATs)," 2008.

[24] S. Liu, R. Zhang-Shen, W. Jiang, J. Rexford, and M. Chiang, "Performance bounds for peer-assisted live streaming," *SIGMETRICS Perform. Eval. Rev.*, vol. 36, no. 1, pp. 313–324, 2008.

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

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

**Olivier Beaumont** received his PhD degree from the University of Rennes in 1999. Between 1999 and 2006, he was assistant professor at Ecole Normale Superieure de Lyon and then at ENSEIRB in Bordeaux. In 2004, he defended his "habilitation a diriger les recherches" and was appointed as Senior Scientist at INRIA in 2007. His research interests focus on the design of parallel and distributed algorithms, overlay networks on large scale heterogeneous platforms and combinatorial optimization.

**Nicolas Bonichon** received his PhD degree from the University of Bordeaux in 2002. He has been holding a position as assistant professor at University of Bordeaux since 2004. His research interests include distributed algorithms, compact data structure, graph drawing and enumerative combinatorics.

**Lionel Eyraud-Dubois** is a Junior Researcher in the CEPAGE team, at INRIA Bordeaux – Sud-Ouest, France. He got his PhD in Computer Science from the Institut National Polytechnique de Grenoble in 2006. His research interests include optimization and approximation algorithms, network modeling, and distributed algorithms.

**Przemysław Uznański** is a PhD student in the CEPAGE team, at INRIA Bordeaux – Sud-Ouest, France. His research interests include enumerative combinatorics and distributed exploration algorithms.

**Shailesh Kumar Agrawal** is a final year BTech-Mtech dual degree student of IIT Kanpur in the department of Computer Science and Engineering. His areas of interest include Computer Networks, Distributed Sytems and Mobile Computing.

## APPENDIX

## VIII. MISSING PROOFS OF SECTION III-A

*Proof (Theorem 3.1):* We prove that the problem of finding an optimal allocation while satisfying the degree constraints (keeping $o_i \leq \left\lceil \frac{b_i}{T} \right\rceil$) is NP-Complete in the strong sense, by reduction to the **3 PARTITION** problem.

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

**3 PARTITION** is well-known to be NP-Complete in the strong sense [26]. Given a particular instance of **3 PARTITION**, let us consider the following instance $\mathcal{I}$ of our problem (see Figure 8), in which all nodes are open.

- The source (the upper node in Figure 8) has outgoing capacity $b_0 = 3pT$;
- $3p$ intermediate nodes (middle nodes in Figure 8), where $\forall 1 \leq i \leq 3p, \ b_i = a_i$;
- $p$ final nodes (lower nodes in Figure 8), where $\forall 3p+1 \leq i \leq 4p, \ b_i = 0$;
- The target throughput to achieve is $T$.

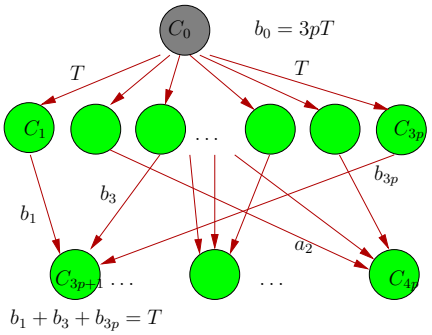


Figure 8. Solution used for the instances used in the reduction.

If a solution to the **3 PARTITION** instance exists, then it is easy to build a solution to $\mathcal{I}$: the source serves all intermediate nodes with rate $T$, and intermediate nodes that correspond to the same set $S_j$ serve a final node $C_{3p+j}$ at their full capacity (see Figure 8).

Conversely, let us assume that there exists a solution to $\mathcal{I}$. We first note that since the total outgoing bandwidth is exactly $4pT$ and $4p$ nodes need to receive the message at rate $T$, it is not possible to waste any bandwidth. Hence, the source necessarily sends data at rate exactly $T$ (*i.e.* the maximal useful possible rate) to $3p$ nodes (*i.e.* the maximal number of clients since $\left\lceil \frac{b_0}{T} \right\rceil = 3p$), and each intermediate node $C_i$ sends data at rate $a_i$ (its maximal rate) to exactly another client (the maximal number of clients since $\left\lceil \frac{a_i}{T} \right\rceil = 1$). On the receiving side, at most $3p$ nodes are served by the source, and the intermediate nodes collectively serve the remaining $p$ nodes (note that nodes served by intermediate nodes may be intermediate nodes themselves, so that the situation is slightly more complicated than depicted in Figure 8). Since no bandwidth is wasted, the sum of the weights of the incoming edges for such a node is exactly $T$. Furthermore, since $\forall i, \quad \frac{T}{4} < a_i < \frac{T}{2}$, there are exactly 3 such incoming edges. It is thus possible to build a solution to the original **3 PARTITION** instance. ■

## IX. MISSING PROOFS OF SECTION IV

### A. Proof of Lemma 4.2

*Proof (Lemma 4.2):*

Let $c$ be an acyclic solution with order $\sigma$ which is not increasing. Then, there exist two indices $x < y$ such that $p = \sigma(x) > q = \sigma(y)$ (and thus $b_p \leq b_q$). We will exhibit another acyclic solution $c'$ with order $\sigma' = \sigma \circ (x, y)$ (where $(x, y)$ denotes the transposition that exchanges $x$ and $y$, which means that the nodes in position $x$ and $y$ are swapped) and whose throughput is not smaller than $c$.

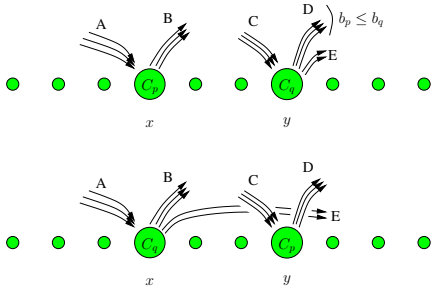


Figure 9. Exchange argument for dominance of increasing solutions

The transformation is depicted on Figure 9. For most indices $i, j$, it is sufficient to set $c'_{\sigma'(i),\sigma'(j)} = c_{\sigma(i),\sigma(j)}$. However, this would break the bandwidth constraint of node $p = \sigma(x)$, and the solution is to give the connections in excess (denoted as E in Figure 9) to node $q = \sigma'(x)$. Since $x < y$, this does not break acyclicity.

Recursively, we can thus transform any acyclic solution into an increasing acyclic solution with at least the same throughput. ■

### B. Proof of Lemma 4.3

*Proof (Lemma 4.3):*

Let $c$ be a solution that achieves $T^*_{ac}(\sigma)$. If there exists a triplet of indices $i, j, k$ that violates *conservativeness*, we can build a solution $c'$ which is conservative

with respect to these indices. Let $\gamma = \min(b_{\sigma(i)} - \sum_{l=i+1}^{k} c_{\sigma(i),\sigma(l)}, c_{\sigma(j),\sigma(k)})$, and set:

$$c'_{\sigma(j),\sigma(k)} = c_{\sigma(j),\sigma(k)} - \gamma$$
$$c'_{\sigma(i),\sigma(k)} = c_{\sigma(i),\sigma(k)} + \gamma.$$

and as in the proof of Lemma 4.2, $\sigma(j)$ will be in charge in $c'$ of the upload toward nodes $\sigma(l)$ with $l > k$ that the node $\sigma(i)$ will no longer be able to feed in $c'$; on all other indices $c$ and $c'$ coincide. It is easy to see that $c'$ is a valid solution of the same throughput, and that the number of triplets of indices violating *conservativeness* is lower in $c'$. Recursively, we create a *conservative* acyclic solution with respect to order $\sigma$, with throughput $T^*_{ac}(\sigma)$. ∎

### C. Proof of Lemma 4.4

*Proof (Lemma 4.4):*
$O$ and $G$ satisfy the following recursive equations:

$$\begin{aligned}
O(\epsilon) &= b_0,\\
G(\epsilon) &= 0,\\
O(\pi\square) &= O(\pi) - T,\\
G(\pi\square) &= G(\pi) + b_{n+j+1},\\
O(\pi\bigcirc) &= O(\pi) + b_{i+1} - \max(0, T - G(\pi)),\\
G(\pi\bigcirc) &= \max(0, G(\pi) - T).
\end{aligned}$$

The values $O$ and $G$ encompass all the capacity constraints of solutions in $\mathcal{C}_\pi$. Indeed, it is easy to see that a coding word $\pi$ is valid for a throughput $T$ if and only if

- for any prefix $\pi'\square$ of $\pi, O(\pi') \geq T$, and
- for any prefix $\pi'\bigcirc$ of $\pi, O(\pi') + G(\pi') \geq T$.

Another parameter that is common to each partial conservative solution of $\mathcal{C}_\pi$ is $W(\pi)$, the amount of transfer going from open nodes to other open nodes. This parameter satisfies the following recursive equations

$$\begin{aligned}
W(\epsilon) &= 0,\\
W(\pi\square) &= W(\pi),\\
W(\pi\bigcirc) &= W(\pi) + \max(0, T - G(\pi)).
\end{aligned}$$

From above, we obtain

$$G(\pi) = b_{n+1} + \ldots + b_{n+j} - i \cdot T + W(\pi) \quad (3)$$
$$O(\pi) = b_0 + b_1 + \ldots + b_i - j \cdot T - W(\pi) \quad (4)$$

and $O(\pi) + G(\pi) = \sum_{k=0}^{|\pi|_\bigcirc} b_k + \sum_{k=n+1}^{n+|\pi|_\square} b_k - |\pi|T$. ∎

### D. Proof of Lemma 4.5

In order to prove Lemma 4.5, we now need to state two preliminary lemmas. The first one shows that this algorithm uses open nodes as late as possible, and is as conservative as possible.

*Lemma 9.1:* Let $\pi_k$ be the value of $\pi$ in Algorithm 2 when the $k$-th open node has just been added. ($|\pi_k|_\bigcirc = k$, and $\pi_k$ ends with a $\bigcirc$).

If $|\pi_k|_\square < m-1$, then for every $\pi'_k$ ending with a $\bigcirc$ such that $|\pi'_k|_\bigcirc = k$, we have

$$W(\pi'_k) \geq W(\pi_k) \quad \text{and} \quad |\pi'_k|_\square \leq |\pi_k|_\square.$$

*Proof:* We prove this lemma by induction on $k$. Clearly the lemma holds true for $k = 0$, since $\pi_0 = \epsilon = \pi'_0$.

Assume now that lemma holds true for $k-1$, and let us decompose the words maximally as follows

$$\pi_k = \pi_{k-1}\square^a\bigcirc \qquad \text{and note} \quad \delta = \pi_{k-1}\square^a,$$
$$\pi'_k = \pi'_{k-1}\square^{a'}\bigcirc.$$

Let $l = |\pi_k|_\square$ and $l' = |\pi'_k|_\square$. From (1) and (2), we get

$$O(\delta) = b_1 + \ldots + b_{k-1} - l \cdot T - W(\pi_{k-1}),$$
$$G(\delta) = b_{n+1} + \ldots + b_{n+l} - (k-1) \cdot T + W(\pi_{k-1}).$$

Since Algorithm 2 chooses $\bigcirc$ (after choosing $\delta$), and $|\delta|_\square < m-1$, we have $O(\delta) < T$ or $O(\delta) + G(\delta) + b_{n+l+1} < 2T$.

Let us first prove by contradiction that $|\pi'_k|_\square \leq |\pi_k|_\square$. Assume that $|\pi'_k|_\square > |\pi_k|_\square$. In this case, there exists $\delta' \sqsubseteq \pi'$ such that $|\delta'| = |\delta|$. By induction assumption, $|\pi_{k-1}|_\square \geq |\pi'_{k-1}|_\square$, which implies that $|\pi'_{k-1}| \leq |\pi_{k-1}| \leq |\delta|$. Hence, $|\delta'|_\bigcirc = |\pi'|_\bigcirc - 1 = k-1$. We can thus compute

$$\begin{aligned}
O(\delta') &= b_1 + \ldots + b_{k-1} - l \cdot T - W(\pi'_{k-1})\\
&\leq b_1 + \ldots + b_{k-1} - l \cdot T - W(\pi_{k-1}) = O(\delta),\\
O(\delta') + G(\delta') &= \sum_{i=1}^{k-1} b_i - l \cdot T + \sum_{i=n+1}^{n+l} b_i - (k-1) \cdot T\\
&= O(\delta) + G(\delta).
\end{aligned}$$

So, either $O(\delta') < T$ or $O(\delta') + G(\delta') + b_{n+l+1} < 2T$. Both lead to a contradiction when we try to continue $\delta'$ with $\square$. This proves that $|\pi'_k|_\square \leq |\pi_k|_\square$.

Let us now prove that $W(\pi'_k) \geq W(\pi_k)$. As $\pi_k$ and $\pi'_k$ end with $\bigcirc$,

$$\begin{aligned}
W(\pi_k) &= W(\pi_{k-1}) + \max(0, T - G(\delta))\\
&= \max(W(\pi_{k-1}), T \cdot k - (b_{n+1} + \ldots + b_{n+l})),\\
W(\pi'_k) &= \max(W(\pi'_{k-1}), T \cdot k - (b_{n+1} + \ldots + b_{n+l'})).
\end{aligned}$$

Since $l' \leq l$ and $W(\pi'_{k-1}) \geq W(\pi_{k-1})$ (the inductive assumption), we have $W(\pi'_k) \geq W(\pi_k)$. ∎

*Lemma 9.2:* Let $\pi_1, \pi_2$ be two conservative partial solutions such that $|\pi_1|_\bigcirc = |\pi_2|_\bigcirc$ and $|\pi_1|_\square = |\pi_2|_\square$. If $W(\pi_1) \leq W(\pi_2)$, then $\forall \omega \in \{\bigcirc, \square\}^*, W(\pi_1\omega) \leq W(\pi_2\omega)$.

*Proof:* To prove the lemma, we only have to consider the cases where $\omega \in \{\bigcirc, \square\}$. The case $\omega = \square$ is trivial since $W(\pi\square) = W(\pi)$.

Let us consider now the case $\omega = \bigcirc$.

$$\begin{aligned} W(\pi_1\bigcirc) &= \max(W(\pi_1), W(\pi_1) + T - G(\pi_1)) \\ &= \max(W(\pi_1), T + i.T - b_{n+1} - \ldots - b_{n+j}) \\ &\leq \max(W(\pi_2), T + i.T - b_{n+1} - \ldots - b_{n+j}) \\ &\leq W(\pi_2\bigcirc). \end{aligned}$$

∎

*Proof (Lemma 4.5):*

The first implication is trivial, since the tests performed at each step of Algorithm 2 ensure that the returned word is always valid.

For the reverse implication, we prove that if Algorithm 2 fails to find a solution, then there does not exist a valid ordering of the nodes with respect to throughput $T$. According to Lemmas 4.2 and 4.3, we only consider encoding words.

Let $\omega$ be the partial solution built by Algorithm 2 (before it failed), and let $i = |\omega|_\bigcirc$ and $j = |\omega|_\square$.

There are four different cases to consider:

- $j < m - 1$ and $\omega$ ends with $\bigcirc$.
  Since Algorithm 2 failed after $\omega$, $O(\omega)+G(\omega) < T$. On the other hand, $O(\omega) \geq b_i$, what implies $b_i < T$ and $\forall k \geq i,\ b_k < T$.
  Let $\pi$ be any encoding word, and let us consider the largest sub-word $\pi' \sqsubseteq \pi$ such that $|\pi'|_\square = |\omega|_\square$. If $|\pi'|_\bigcirc < |\omega|_\bigcirc$, then there exists a word $\rho \sqsubseteq \pi$ such that $|\rho|_\bigcirc = |\omega|_\bigcirc$ and $|\rho|_\square > |\omega|_\square$. Since this violates the conclusions of Lemma 9.1, $\pi$ is not valid.
  If $|\pi'|_\bigcirc \geq |\omega|_\bigcirc$, then

$$\begin{aligned} O(\pi') + G(\pi') &= O(\omega) + G(\omega) + \sum_{k=i+1}^{|\pi'|_\bigcirc} (b_k - T) \\ &\leq O(\omega) + G(\omega) < T. \end{aligned}$$

  In conclusion, $O(\pi') < T$ and thus $\pi$ is not valid.
- $j \leq m - 1$ and $\omega$ ends with $\square$.
  Because of the test at line 12, this implies that the last $\square$ was added by the instruction on line 4, and thus $|\omega|_\bigcirc = n$.
  Let $\pi$ be an encoding word. We can decompose $\omega$ and $\pi$ as $\omega' \bigcirc \square^a$ and $\pi' \bigcirc \square^b$, and we can apply Lemma 9.1 to words $\omega'\bigcirc$ and $\pi'\bigcirc$

$$W(\omega) = W(\omega') \leq W(\pi') = W(\pi)$$

  Since $|\omega|_\bigcirc = n$ and since Algorithm 2 failed, then either $O(\omega) + G(\omega) < T$ or $O(\omega\square) < 0$. In both cases, $O(\omega) < T$, and since $O(\omega) = O - jT - W(\omega)$, we get $O < mT + W(\pi)$, and thus $O(\pi) < 0$. Hence $\pi$ is not valid.
- $j = m$. The main argument is that the bandwidth of the remaining open nodes is lower than that of the last guarded node. Since Algorithm 2 chose the last guarded node at some point (line 11), we have $b_{i+1} \leq b_{n+m}$.
  The failure of the algorithm implies $O(\omega)+G(\omega) < T$. Let $\omega = \omega'\alpha$. We know that $O(\omega')+G(\omega') \geq T$, and also:

$$\begin{aligned} O(\omega) + G(\omega) &= O(\omega') + G(\omega') - T + b_i && \text{if } \alpha = \bigcirc, \\ O(\omega) + G(\omega) &= O(\omega') + G(\omega') - T + b_{n+m} && \text{if } \alpha = \square. \end{aligned}$$

  So either $b_{n+m} < T$ or $b_i < T$. In both cases, we have

$$b_n \leq b_{n-1} \leq \ldots \leq b_{i+1} < T.$$

  Let $\pi = \pi'\beta$ be any encoding word. If $\beta = \bigcirc$, then

$$\begin{aligned} O(\pi') + G(\pi') &= b_0 + O - b_n - (n-1)T + G - mT \\ &= O(\omega) + G(\omega) + \sum_{k=i+1}^{n-1} (b_k - T) \\ &\leq O(\omega) + G(\omega) < T \end{aligned}$$

  Hence $\pi$ is not valid. Otherwise, $\beta = \square$, and $O(\pi') + G(\pi') = b_0 + O - nT + G - b_{n+m} - (m-1)T$. Since $b_{n+m} \geq b_n$, we get the same conclusion.
- $j = m - 1$ and $\omega$ ends with $\bigcirc$. The main argument is that the last guarded node can be delayed: minimizing waste is not so important since only open nodes remain to be fed. Just like in the first case, we have $\forall k \geq i, b_k < T$. Let us decompose $\omega$ as $\omega = \omega'\bigcirc^a$ $(a \geq 0)$.
  We begin by showing that words $\pi(x) = \omega'\square \bigcirc^x \square\bigcirc^{a-x}$ are invalid for throughput $T$. The following lemma shows that it is possible to consider only words where the last $\square$ is followed only by $\bigcirc$ with smaller bandwidth.
  *Lemma 9.3:* If word $\pi = \pi_1\square \bigcirc \bigcirc^a$ is a valid word in which the last $\square$ has bandwidth $g$, the following $\bigcirc$ has bandwidth $o$, and $o \geq g$, then the word $\pi' = \pi_1 \bigcirc \square\bigcirc^a$ is also valid.

*Proof:* Let $G = O(\pi_1), R = G(\pi_1)$ and $\pi_2 = \bigcirc^a$. Since $\pi$ is valid, we have $O \geq T$ and $O - T + G + r \geq T$. We can thus bound $O(\pi_1 \bigcirc)$

$$\begin{aligned} O(\pi_1 \bigcirc) &= O + o - \max(T - G, 0) \\ &= \min(O + o + G - T, O + o) \geq T. \end{aligned}$$

This ensures that $\pi_1 \bigcirc \square$ is a valid sequence.
Since $\pi_2$ is composed only of $\bigcirc$, and $O(\pi_1 \square \bigcirc) + G(\pi_1 \square \bigcirc) = O(\pi_1 \bigcirc \square) + G(\pi_1 \bigcirc \square)$, $\pi_1 \bigcirc \square \pi_2$ is a valid sequence. ∎
So if $\pi(x)$ is valid, we can iteratively use Lemma 9.3 to prove the existence of a valid $\pi(y)$ in which the last $\square$ is followed by a $\bigcirc$ with smaller upload. If $y < a$, since Algorithm 2 at that point chose $\bigcirc$ instead of $\square$, we know that $O(\omega' \square \bigcirc^y) < T$ and $\pi(y)$ is invalid. If $y = a$, then $\pi(y) = \omega \square$, which is invalid because Algorithm 2 failed.
Consider now any encoding word $\pi$. Let $\pi = \pi_1 \pi_2 \square \bigcirc^k$ be the decomposition with minimal $\pi_1$ having $|\pi_1|_\bigcirc = |\omega'|_\bigcirc$ (applying Lemma 9.1 we have $|\pi_1|_\square \leq |\omega'|_\square = m - 2$, so decomposing is always possible).
For any word $\delta$ we have

$$W(\delta \bigcirc \square) = W(\delta \bigcirc) = W(\delta) + \max(0, T - G(\delta)) \geq$$

$$\geq W(\delta \square) + \max(0, T - G(\delta \square)) = W(\delta \square \bigcirc).$$

We can apply it to word $\pi$

$$W(\pi_1 \pi_2) \geq W(\pi_1 \square^{|\pi_2|_\square} \bigcirc^{|\pi_2|_\bigcirc}).$$

Furthermore, since Lemma 9.1 applies to $\pi_1$ and $\omega'$:

$$W(\pi_1) \geq W(\omega').$$

so by Lemma 9.2, (since $|\pi_1|_\bigcirc = |\omega'|_\bigcirc$, $|\pi_1 \pi_2|_\square = m - 1 = |\omega' \square|_\square$)

$$W(\pi_1 \square^{|\pi_2|_\square} \bigcirc^{|\pi_2|_\bigcirc}) \geq W(\omega' \square \bigcirc^{|\pi_2|_\bigcirc}).$$

Composing it, we have

$$W(\pi_1 \pi_2) \geq W(\omega' \square \bigcirc^{|\pi_2|_\bigcirc}) \text{and}$$

$$\forall x, W(\pi_1 \pi_2 \square \bigcirc^x) \geq W(\omega' \square \bigcirc^{|\pi_2|_\bigcirc} \square \bigcirc^x).$$

So if $\pi = \pi_1 \pi_2 \square \bigcirc^k$ is valid, then $\omega' \square \bigcirc^{|\pi_2|_\bigcirc} \square \bigcirc^k$ is also valid. But we proved previously that no such solution can exist. Thus, we reached a contradiction.

∎

### E. Proof of Lemma 4.6

*Proof (Lemma 4.6):*
Since guarded nodes can only upload to open nodes, and open nodes always receive from the earliest guarded node available, every guarded node uploads to a consecutive interval of open nodes. So at most 2 nodes will be partially fed by a specific guarded node: the first and the last one of the interval (see first example of Figure 10).

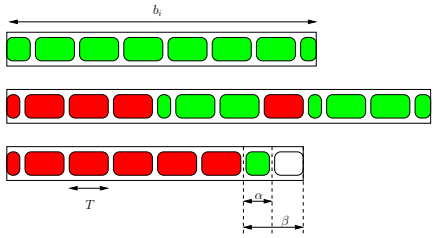


Figure 10. 3 examples of upload node repartition. A guarded node feeds at most 2 nodes partially (first example). An open node that is the first to feed the last guarded node (second example). General case for an open node (last example).

Let us now consider an open node $i$. Because Algorithm 2 rather chooses guarded nodes when it is possible, as long as there is enough open bandwidth available, node $i$ will feed a consecutive interval of guarded nodes. When the amount of open upload available gets low, there are two cases to consider:

- $i$ is the earliest open node that feeds the last guarded node.
  The sequence of nodes fed by $i$ first consists in a sequence of guarded nodes, then a sequence of open nodes, then the last guarded node and another sequence of open nodes (see second example of Figure 10). Since conservatism implies that $G(\pi \bigcirc) = 0$ after feeding an open node from node $i$, a partially fed open node can only take place as the first node after guarded nodes. Hence, the only nodes partially fed by $i$ are the first one, the last one and the opening nodes of the 2 open sequences. In total, at most 4 nodes are partially fed by node $i$.
- Otherwise (see last example of Figure 10), Algorithm 2 feeds guarded nodes with the upload of $i$ as long as there is enough bandwidth. At some point, $O + G + g_{\text{next}} < 2T$, where $g_{\text{next}}$ is the bandwidth of the next guarded node to be fed. Let $\beta$ be the remaining bandwidth of node $i$ at that point. By the definition of $O$, $\beta \leq O$. At this moment, Algorithm 2 decides to switch to open nodes. Open nodes are fed using guarded bandwidth at first. If any open node is fed using $\alpha = T - G$

upload from $i$, the remaining upload of $i$ is equal to $\beta - \alpha \leq O + G - T \leq T - g_{\text{next}} \leq T$. Thus, the next node fed by $i$ uses all the remaining bandwidth of node $i$. Hence, node $i$ feeds partially at most 3 nodes: the first node, one open node and the last node.

■

## X. Missing proofs of Section V

### A. Proof of Theorem 5.2

*Proof (Theorem 5.2):* Basically the algorithm works in two steps. The first step of this algorithm consists in executing Algorithm 1 until the smallest index $i_0$ such that $S_{i_0-1} < i_0 T$ (we recall the notation $S_k = \sum_{i=0}^{k} b_i$). If there is no such index, then Algorithm 1 outputs a valid solution of throughput $T$. Otherwise, the result is a partial solution in which all nodes up to $i_0 - 1$ are served at rate $T$, and nodes with indexes $i_0$ or larger do not send nor receive anything. In what follows, we will call such a solution a $(i_0-1)$-partial solution (see Figure 11). The second step of the cyclic algorithm involves building successive $i$-partial solutions for $i$ equal to $i_0 + 1$, ..., until $n$ by applying local changes on the previous partial solution. In the complete solution, the degree of nodes $C_i$ with $i < i_0$ is increased by at most one during the second step, which leads to an out degree of at most $o_i \leq \left\lceil \frac{b_i}{T} \right\rceil + 2$, and the out degree of nodes $C_i$ with $i \geq i_0$ is at most 4.

Figure 11 shows a partial solution produced at the end of the first step and Figure 12 shows the complete solution.

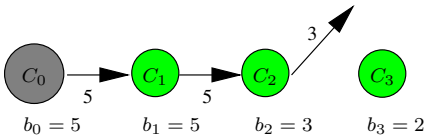


Figure 11. After applying Algorithm 1 with $T = 5$, there is not enough bandwidth to feed $C_3$. In this case $i_0 = 3$.

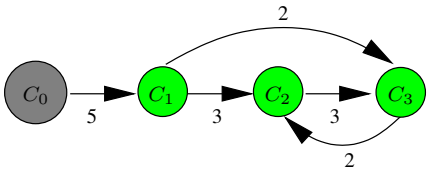


Figure 12. A solution computed from the 2-partial solution of Figure 11.

The first part of this algorithm consists in executing Algorithm 1 until the smallest index $i_0$ such that $S_{i_0-1} < i_0 T$ (we recall the notation $S_k = \sum_{i=0}^{k} b_i$). If there is no such index, then Algorithm 1 outputs a valid solution of throughput $T$. Otherwise, the result is a partial solution in which all nodes up to $i_0 - 1$ are served at rate $T$, and nodes with indexes $i_0$ or larger do not send nor receive anything. In what follows, we will call such a solution a $(i_0-1)$-partial solution (see Figures 11 and 14). The next part of the cyclic algorithm involves building successive $i$-partial solutions for $i$ equal to $i_0$, $i_0 + 1$, ..., until $n$.

For all values $i \geq i_0$, let $M_i = iT - S_{i-1}$ be the missing flow at node $C_i$ when the bandwidth of all previous nodes are completely used. In a $i$-partial solution, it is compulsory that $C_i$ sends a flow $M_i$ toward previous nodes so that the total input and output flow rates are equal. Let $R_i = b_i - M_i$ be the remaining capacity at $C_i$ in such a solution. These definitions imply the following property: $R_i + M_{i+1} = b_i - iT + S_{i-1} + (i+1)T - S_i = T$. By induction on $i$, we can prove that it is possible to build successive $i$-partial solutions such that:

- (P1) $c_{i,i-1} + c_{i-1,i} = T$,
- (P2) the out-degree of $C_i$ is at most 2,
- (P3) the out-degree of $C_{i-1}$ is at most 3,
- (P4) the remaining available bandwidth of $C_i$ is $R_i$.

For the sake of simplicity, we first assume $n > i_0$ (the particular and simpler case $n = i_0$ will be discussed later).

*a) Initial case, $i = i_0 < n$:* We start from the $(i-1)$-partial solution built using Algorithm 1 (see Figure 14) and we build a $(i+1)$-partial solution. In this $(i-1)$-partial solution, $C_i$ already receives a flow $T - M_i$ from a set of nodes $\mathcal{A}$ (that all receive the message at rate $T$). We select an arbitrary edge $(C_u, C_v)$ with capacity at least $M_i$[1]. Since $n \geq i + 1$, we set $\alpha = \max(0, M_{i+1} - M_i)$ and $\beta = M_{i+1} - \alpha$ and make the following modifications (depicted in Figure 13 and applied on an example in Figure 15):

- Flow $\alpha$ goes from $\mathcal{A}$ to $C_{i+1}$ instead of $C_i$;
- Flow $M_i$ goes from $u$ to $C_i$ instead of $C_v$;
- $C_i$ sends a flow $R_i + \beta$ to $C_{i+1}$;
- $C_i$ sends a flow $M_i - \beta$ to $C_v$;
- $C_{i+1}$ sends a flow $\beta$ to $C_v$ and a flow $\alpha$ to $C_i$.

The choices of $\alpha$ and $\beta$ ensure that the flow on all edges remains positive, that no node exceeds its outgoing bandwidth and that $\alpha + \beta = M_{i+1}$, so that $R_i + \beta + \alpha = T$. Hence, all the nodes receive a flow at rate $T$ from the source node. Therefore, we have built a $(i_0 + 1)$-partial solution, with $c_{i_0+1,i_0} + c_{i_0,i_0+1} = T$, and the outdegrees

[1]The fact that $T \leq b_0$ ensures that $C_1$ necessarily receives $T$ from the source node, hence such an edge always exists.

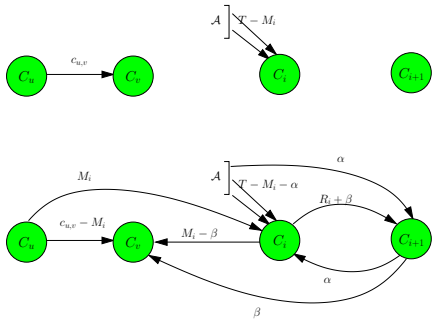


Figure 13. Modifications for the initial case.

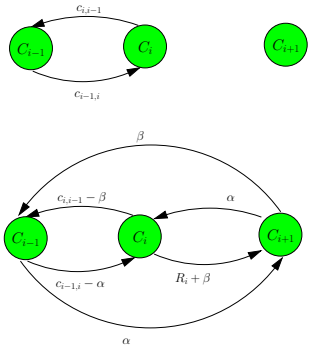


Figure 16. Modifications when adding $C_{i+1}$ in the inductive case.

of nodes $i_0$ and $i_0+1$ are both equal to 2. Finally, $C_{i+1}$ sends a flow $\alpha+\beta$, and thus has $b_{i+1} - M_{i+1} = R_{i+1}$ remaining available bandwidth.

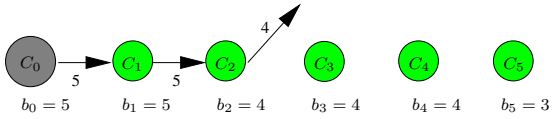


Figure 14. Example of a 2-partial solution computed by Algorithm 1 initial case. $T = 5$, $i_0 = 3$, $M_3 = 1$.

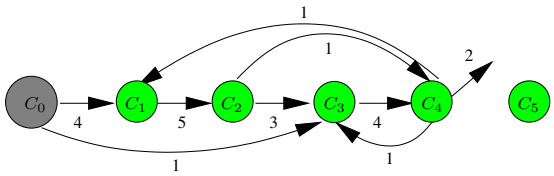


Figure 15. The 4-partial solution computed after the application of the modification of the initial case. On this example we took $C_u = C_0$ and $C_v = C_1$.

*b) Induction:* Let us assume that we have built a $i$-partial solution satisfying properties (P1)-(P4). $C_{i+1}$ is inserted as follows (see Figure 16):

- $C_i$ uses all its remaining bandwidth $R_i$ to send data to $C_{i+1}$;
- Part $\alpha$ of the flow going from $C_{i-1}$ to $C_i$ now goes through $C_{i+1}$;
- Part $\beta$ of the flow going from $C_i$ to $C_{i-1}$ now goes through $C_{i+1}$.

Figure 17 shows a concrete example of this modification.

Once again, we set $\alpha = \max(0, M_{i+1} - c_{i,i-1})$ and $\beta = M_{i+1} - \alpha = \min(M_{i+1}, c_{i,i-1})$. It is easy to check that this choice ensures that $\alpha \leq c_{i-1,i}$ and that nodes $i$ and $i-1$ receive a flow $T$ from the source node (in the same way as in the original $i$-partial solution, with just a diversion through $C_{i+1}$). Furthermore, $C_{i+1}$ receives a flow $\alpha$ from $C_{i-1}$, a flow $c_{i-1,i} - \alpha$ from $C_{i-1}$ through $C_i$, and a flow $c_{i,i-1}$ from node $i$. Note that these three flows are compatible, since the total flow going from $C_i$ to $C_{i+1}$ is $c_{i-1,i} - \alpha + c_{i,i-1} = T - \alpha = T - M_{i+1} + \beta = R_i + \beta$, *i.e.* the capacity of the edge.

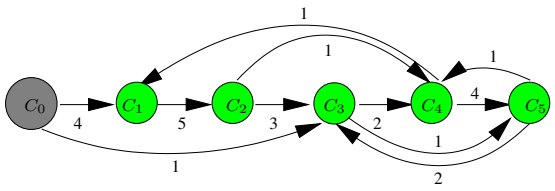


Figure 17. Complete solution computed after the application of the inductive case.

This solution is thus a $(i+1)$-partial solution, with $C_{i+1}$ having out-degree 2 and $C_i$ having out-degree at most 3 (one edge has been added to the previous $i$-partial solution). This concludes the proof.

*c) If $i_0 = n$:* 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 $R_{i_0}$ is ignored (see Figure 12).

*d) Overall solution:* In the solution obtained by recursively applying the above procedure, the actual out-degree $o_i$ of $C_i$ is at most $\max(\lceil \frac{b_i}{T} \rceil + 2, 4)$:

- In the $(i_0 - 1)$-partial solution obtained at the end of algorithm 1, $o_i \leq \lceil \frac{b_i}{T} \rceil + 1$;
- During the initialization phase, the outdegree of exactly two clients increases by 1;
- When adding $C_{i+1}$, the outdegree of node $C_{i-1}$ is increased by 1, and it was at most 3.

■

## XI. Missing proofs of Section VI

### A. Proof of Theorem 6.1

*Proof (Theorem 6.1):*

Let $I$ be an instance of size $n$ without guarded nodes. From Section III we know that

$$T^*_{ac} = \min(b_0; \frac{b_0 + O - b_n}{n}).$$

From Lemma 5.1 we have

$$T^* \leq \min(b_0; \frac{b_0 + O}{n}).$$

If $T^*_{ac} = b_0$ then it is also the case for $T^*$ and the result holds. Else we have

$$\frac{T^*_{ac}}{T^*} \geq \frac{b_0 + O - b_n}{b_0 + O} = 1 - \frac{b_n}{b_0 + O}.$$

Because of the ordering of nodes we have $O \geq nb_n$. This concludes the proof. ■

### B. Proof of Theorem 6.2

We start this proof by characterizing a special class of instances which are the worst possible cases for the acyclic throughput. An instance is said to be *homogeneous* if all open nodes except the source have the same throughput $o$ and all guarded nodes have the same throughput $g$. An instance is said to be *tight* if $b_0 = \frac{b_0+O+G}{n+m} = T^*$ (*i.e.* if no bandwidth can be wasted in the optimal cyclic solution). The instance of Figure 1 is tight but not homogeneous and the instance of Figure 6 is tight and homogeneous.

*Lemma 11.1:* Let $\alpha > 0$. If for every tight homogeneous instance, $\frac{T^*_{ac}}{T^*} \geq \alpha$, then for every instance $\frac{T^*_{ac}}{T^*} \geq \alpha$.

*Proof:* To prove this lemma we will show that given an instance, we can associate with it a tight homogeneous instance with the same optimal throughput $T^*$ and with no greater optimal acyclic throughput $T^*_{ac}$.

First, if the instance is such that $\frac{b_0+O+G}{n+m} > T^*$, by reducing the throughput of the guarded nodes it is possible to make this inequality an equality. This transformation does not change the optimal throughput $T^*$ and any acyclic solution for the transformed instance is also an acyclic solution for the original one.

Consider now a non-homogeneous instance $I$ such that $\frac{b_0+O+G}{n+m} = T^*$. Let $I'$ be the homogeneous instance obtained from $I$ as follows: $b'_0 = T^*, b'_i = o = (N + b_0 - T^*)/n$ for $i \in [\![1, n]\!]$ and $b'_i = g = M/m$ for $i \in [\![n+1, n+m]\!]$, where $b'_i$ is the throughput of the node $C_i$ in $I'$. Clearly $I$ and $I'$ have the same optimal throughput $T^*$ and $I'$ is tight and homogeneous. Observe that since nodes of same color are ordered in the non-increasing order of their throughput, $\forall k \in [\![0, n]\!], \sum_{i=0}^{k} b_i \geq \sum_{i=0}^{k} b'_i$ and $\forall k \in [\![n+1, n+m]\!], \sum_{i=1}^{k} b_i \geq \sum_{i=1}^{k} b'_i$. Hence, any acyclic scheme of $I'$ can be turned into a scheme where $I$ communications previously ensured by the $k$-th open (resp. guarded) node in $I'$ are now ensured by the $k$ first open (resp. guarded) nodes in $I$. The resulting scheme is acyclic and achieves the same throughput. ■

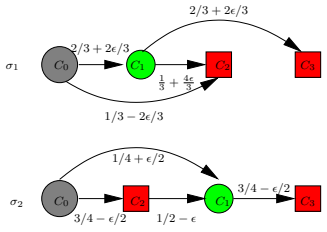


Figure 18. Optimal acyclic schemes of $\sigma_1$ and $\sigma_2$.

*Proof (Theorem 6.2):*

Let us first show that the ratio 5/7 can be reached. For this purpose, let us consider the following instance (see Figure 18) consisting of one source of throughput 1, one open node of throughput $b_1 = 1 + 2\epsilon$ and two guarded nodes with throughput of $b_2 = b_3 = 1/2 - \epsilon$ each. For this instance, $T^* = 1$ (see Lemma 5.1). There also exist 3 increasing orderings $\sigma_1 = 0123, \sigma_2 = 0213$ and $\sigma_3 = 0231$. Ordering $\sigma_1$ achieves a throughput of $T^*_{ac}(\sigma_1) = (2/3).(1 + \epsilon)$ and ordering $\sigma_2$ achieves a throughput of $T^*_{ac}(\sigma_2) = 3/4 - \epsilon/2$ (see Figure 18). The throughput of the last ordering is always smaller than the maximum of the two previous ones. When $\epsilon = 1/14$, orderings $\sigma_1$ and $\sigma_2$ achieve the same throughput $T^*_{ac} = 5/7$.

Let us now prove that for any instance $\frac{T^*_{ac}}{T^*} \geq \frac{5}{7}$. Without loss of generality, we can consider only tight homogeneous instances. We can also assume that $n \geq 1$, $m \geq 2$ and $n + m \geq 4$ since other cases are trivial or have been considered above.

Let us consider the following two words

$$\omega_1(n, m) = \bigcirc\square^{\alpha_1} \bigcirc \square^{\alpha_2} \ldots \bigcirc \square^{\alpha_n},$$

$$\omega_2(n, m) = \square \bigcirc^{\beta_1} \square \bigcirc^{\beta_2} \ldots \square \bigcirc^{\beta_n}.$$

where $\alpha_i = \lfloor i \cdot \frac{m}{n} \rfloor - \lfloor (i-1) \cdot \frac{m}{n} \rfloor$ and $\beta_i = \lceil i \cdot \frac{n}{m} \rceil - \lceil (i-1) \cdot \frac{n}{m} \rceil$.

As observed in Section IV, these words encode increasing orders on vertices. To conclude the proof we only have to show that at least one of these two words encodes a valid scheme of throughput $5/7$

$$\max(T^*_{ac}(\omega_1(n, m)), T^*_{ac}(\omega_2(n, m))) \geq 5/7.$$

Recall that we consider tight homogeneous instances with $b_0 = 1$ such that $n \geq 1$, $m \geq 2$ and $n + m \geq 4$. Without loss of generality, we can also assume that $b_0 = 1, b_0 + O \geq m$ and $b_0 + O + G = n + m$. Hence for some $0 \leq \Delta \leq n$, the bandwidth of each open node is $o = \frac{m-1+\Delta}{n}$ and the bandwidth of each guarded node is $g = \frac{n-\Delta}{m}$. To show that $\max(T^*_{ac}(\omega_1(n,m)), T^*_{ac}(\omega_2(n,m))) \geq 5/7$, we will show a more precise statement

$$\begin{aligned} &\text{if } o \geq 1, T^*_{ac}(\omega_1(n,m)) \geq 5/7 \\ &\text{otherwise } T^*_{ac}(\omega_2(n,m)) \geq 5/7. \end{aligned} \tag{5}$$

Let us start with two additional technical lemmas.

*Lemma 11.2:* For a tight homogeneous instance, a word $\omega$ is valid for throughput $T$ if and only if

- **(c1)** $\forall_{\pi \bigcirc \sqsubseteq \omega} b_0 + o \cdot |\pi|_{\bigcirc} + g \cdot |\pi|_{\square} - |\pi \bigcirc| \cdot T \geq 0$
- **(c2)** $\forall_{\pi' \bigcirc \sqsubseteq \pi \square \sqsubseteq \omega} b_0 + o \cdot |\pi|_{\bigcirc} + g \cdot |\pi'|_{\square} - |\pi \square|_{\square} \cdot T - |\pi' \bigcirc|_{\bigcirc} \cdot T \geq 0$

*Proof:* As shown in Section IV, a word $\omega$ is valid for a throughput $T$ if and only if

$$\begin{aligned} &\text{for any prefix of } \omega \text{ of the form } \pi\square, G(\pi) \geq T, \\ &\text{for any prefix of } \omega \text{ of the form } \pi\bigcirc, G(\pi) + R(\pi) \geq T. \end{aligned}$$

For homogeneous instances, the second condition can be written as

$$\forall_{\pi \bigcirc \sqsubseteq \omega} b_0 + o \cdot |\pi|_{\bigcirc} + g \cdot |\pi|_{\square} - |\pi \bigcirc| \cdot T \geq 0$$

which is exactly (c1).

And the first condition is

$$\forall_{\pi \square \sqsubseteq \omega} b_0 + o \cdot |\pi|_{\bigcirc} - |\pi \square|_{\square} \cdot T - W(\pi) \geq 0.$$

From the recursive equations which define $W$ and $G$, we can deduce

$$W(\pi\bigcirc) = \max(W(\pi), |\pi \bigcirc|_{\bigcirc} \cdot T - (b_{n+1} + \ldots + b_{n+|\pi|_{\square}})).$$

Together with $W(\pi\square) = W(\pi)$, this implies

$$W(\pi) = \max_{\pi' \bigcirc \sqsubseteq \pi} \{|\pi' \bigcirc|_{\bigcirc} \cdot T - (b_{n+1} + \ldots + b_{n+|\pi'|_{\square}})\},$$

hence the first condition can be rewritten to

$$\begin{aligned} &\forall_{\pi' \bigcirc \sqsubseteq \pi \square \sqsubseteq \omega}, \\ &b_0 + o \cdot |\pi|_{\bigcirc} g \cdot |\pi'|_{\square} - |\pi\square|_{\square} \cdot T - |\pi' \bigcirc|_{\bigcirc} \cdot T \geq 0. \end{aligned}$$

■

*Lemma 11.3:* If $\omega$ is an encoding word for a valid solution of an homogeneous instance with $(b_0 = b'_0, o = o', g = g')$ with throughput $T$, and $\omega$ is also encoding a valid solution of an homogeneous instance with $(b_0 = b''_0, o = o'', g = g'')$ with throughput $T$, and $0 \leq \lambda_1, \lambda_2 \leq 1$ are such that $\lambda_1 + \lambda_2 = 1$, then $\omega$ is also a valid solution for homogeneous instance with $(b_0 = \lambda_1 \cdot b'_0 + \lambda_2 \cdot b''_0, o = \lambda_1 \cdot o' + \lambda_2 \cdot o', g = \lambda_1 \cdot g' + \lambda_2 \cdot g'')$ with throughput $T$.

*Proof:* For fixed $\pi\bigcirc$, we can write (c1) as

$$\begin{aligned} &b''_0 + o'' \cdot |\pi|_{\bigcirc} + g'' \cdot |\pi|_{\square} - |\pi \bigcirc| \cdot T = \\ &= \lambda_1 (b_0 + o \cdot |\pi|_{\bigcirc} + g \cdot |\pi|_{\square} - |\pi \bigcirc| \cdot T) + \\ &+ \lambda_2 (b'_0 + o' \cdot |\pi|_{\bigcirc} + g' \cdot |\pi|_{\square} - |\pi \bigcirc| \cdot T) \geq 0. \end{aligned}$$

Condition (c2) is proved analogously. ■

Now let us go back to the proof of statement (5).

Since when $m > n$, it is impossible to have $o < 1$, we need to consider only 3 cases

- $m \geq n + 1$ and $o \geq 1$,
- $m \leq n$ and $o \geq 1$,
- $m \leq n$ and $o \leq 1$.

Using Lemma 11.3, we can eliminate the parameter $\Delta$ from each of the cases, reducing each of them to two extreme cases:

- $m \geq n + 1$ and $o \geq 1$
  - **(A1)** $o = \frac{m-1}{n}$, $g = \frac{n}{m}$,
  - **(A2)** $o = \frac{n+m-1}{n}$, $g = 0$.
- $m \leq n$ and $o \geq 1$
  - **(B1)** $o = 1$, $g = \frac{m-1}{m}$,
  - **(B2)** $o = \frac{n+m-1}{n}$, $g = 0$.
- $m \leq n$ and $o \leq 1$
  - **(C1)** $o = \frac{m-1}{n}$, $g = \frac{n}{m}$,
  - **(C2)** $o = 1$, $g = \frac{m-1}{m}$.

We now check for each case that the appropriate word ($\omega_1(n,m)$ or $\omega_2(n,m)$) satisfies the conditions **(c1)** and **(c2)** of Lemma 11.2.

*Lemma 11.4:* In cases **(A2)** and **(B2)**, $T^*_{ac} \geq 5/7$.

*Proof:* Merging cases **(A2)** and **(B2)** together, we consider the following instance

$$o = \frac{n+m-1}{n}, g = 0,$$

($o \geq 1$ obviously holds), and the encoding word $\omega_1(n,m)$.

It is enough to verify condition (c1), because open bandwidth is the only available bandwidth in this case. If we denote $|\pi|_{\bigcirc}$ as $i$, $0 \leq i < n$, (c1) becomes then

$$\forall_{0 \leq i < n} 1 + \frac{n+m-1}{n} \cdot i - \left(i + 1 + \left\lfloor \frac{m}{n} i \right\rfloor\right) \cdot \frac{5}{7} \geq 0.$$

Since $\lfloor \frac{m}{n} i \rfloor \leq \frac{m}{n} i$, it is enough to prove

$$1 + \frac{n+m-1}{n} \cdot i - \left(i + 1 + \frac{m}{n} i\right) \cdot \frac{5}{7} \geq 0$$

which simplifies to

$$\frac{2}{7}+\frac{2}{7}i+\frac{2}{7}\frac{m}{n}i-\frac{i}{n}\geq 0$$
$$2+(2n+2m-7)i\geq 0$$

which holds, since $n+m\geq 4$. ∎

*Lemma 11.5:* In cases **(B1)** and **(C2)**, $T^*_{ac}\geq 5/7$.

*Proof:* Increasing $n$ in those cases only results in adding open nodes with bandwidth $1\geq\frac{5}{7}$. So it is enough to prove the two conditions for $n=m$.

So we now assume $n=m$. If $m\geq 4$, then $g\geq\frac{3}{4}$, and

$$\omega_1(n,n)=(\bigcirc\square)^n$$
$$\omega_2(n,n)=(\square\bigcirc)^n,$$

and every node is able to feed the next node.

If $m\leq 3$, we can easily verify that the words $\omega_1(2,2)$ and $\omega_2(2,2)$ are valid for the case $o=1,g=\frac{1}{2}$ with throughput $T=\frac{5}{7}$, and that the words $\omega_1(3,3)$ and $\omega_2(3,3)$ are valid for the case $o=1,g=\frac{2}{3}$ with throughput $T=\frac{5}{7}$. ∎

*Lemma 11.6:* In case **(A1)**, $T^*_{ac}\geq 5/7$.

*Proof:* Let us check condition (c1). For the sake of readability, let us denote by $i$ the value $|\pi|_\bigcirc$ ($0\leq i<n$). Condition (c1) can be rewritten to

$$1+i\cdot\frac{m-1}{n}+\left\lfloor i\cdot\frac{m}{n}\right\rfloor\cdot\frac{n}{m}\geq\frac{5}{7}\left(1+i+\left\lfloor i\cdot\frac{m}{n}\right\rfloor\right)$$

which is equivalent to

$$\frac{2}{7}+i\left(\frac{m-1}{n}-\frac{5}{7}\right)\geq\left\lfloor i\cdot\frac{m}{n}\right\rfloor\cdot\left(\frac{5}{7}-\frac{n}{m}\right).$$

Since $\frac{m-1}{n}\geq 1$, the left side is always positive. We can safely assume that $\frac{5}{7}\geq\frac{n}{m}$ (otherwise, the right side is negative). And since $\lfloor i\cdot\frac{m}{n}\rfloor<i\cdot\frac{m}{n}$, it is enough to prove

$$\frac{2}{7}+i\left(\frac{m-1}{n}-\frac{5}{7}\right)\geq i\cdot\frac{m}{n}\cdot\left(\frac{5}{7}-\frac{n}{m}\right).$$

Simplifying, we get

$$\frac{2}{7}+i\left(\frac{2}{7}+\frac{2}{7}\cdot\frac{m}{n}-\frac{1}{n}\right)\geq 0.$$

Since $\frac{2}{7}+\frac{2}{7}\cdot\frac{m}{n}\geq\frac{2}{7}+\frac{2}{7}\cdot\frac{7}{5}\geq\frac{1}{2}\geq\frac{1}{n}$, condition (c1) holds.

Let us now check condition (c2). We denote $|\pi|_\bigcirc=i$ and $|\pi'|_\bigcirc=j$ (it is enough to consider the longest such $\pi'$), $0\leq j<i\leq n$.

$$1+i\cdot\frac{m-1}{n}+\left\lfloor\frac{m}{n}\cdot j\right\rfloor\cdot\frac{n}{m}-\left\lfloor\frac{m}{n}\cdot i\right\rfloor\cdot\frac{5}{7}-(j+1)\cdot\frac{5}{7}\geq 0.$$

Simplifying, we get

$$\left(\frac{2}{7}+\left\lfloor\frac{m}{n}\cdot j\right\rfloor\frac{n}{m}-\frac{5}{7}j\right)+\frac{5}{7}\left(\frac{m}{n}\cdot i-\left\lfloor\frac{m}{n}\cdot i\right\rfloor\right)+\frac{i}{n}\left(\frac{2}{7}m-1\right)\geq 0.$$

Now, we can also use this

$$\forall_{x\geq 0}\lfloor x\rfloor=\lfloor x\rfloor\cdot\frac{\lfloor x\rfloor+1}{\lfloor x\rfloor+1}\geq\frac{x\cdot\lfloor x\rfloor}{\lfloor x\rfloor+1}.$$

So we have

$$\lfloor\frac{m}{n}\cdot j\rfloor\geq\frac{m}{n}j\cdot\frac{\lfloor\frac{m}{n}\cdot j\rfloor}{\lfloor\frac{m}{n}\cdot j\rfloor+1}\geq\frac{m}{n}j\cdot\frac{j}{j+1}.$$

Using this, condition (c2) holds if

$$\left(\frac{2}{7}+j\cdot\frac{j}{j+1}-\frac{5}{7}j\right)+\frac{5}{7}\left(\frac{m}{n}\cdot i-\left\lfloor\frac{m}{n}\cdot i\right\rfloor\right)+\frac{i}{n}\left(\frac{2}{7}m-1\right)\geq 0.$$

The case $m\leq 3$ can be checked separately ($m=3,n=2,o=1,r=\frac{2}{3}$). When $m\geq 4$, we have $\frac{2}{7}m-1>0$, and it remains to prove that

$$\frac{2}{7}+j\cdot\frac{j}{j+1}-\frac{5}{7}j\geq 0$$

For $j\in\{0,1,2,3\}$ it can be checked, and for $j\geq 4$ the following inequality allows us to conclude

$$j\cdot\frac{j}{j+1}\geq\frac{5}{7}j.$$

∎

*Lemma 11.7:* In case **(C1)**, $T^*_{ac}\geq 5/7$.

*Proof:*

If we denote $i=|\pi|_\square$ (the case $i=0$ is trivial, so we can assume $0<i<m$), condition (c1) can be written as

$$1+\frac{n}{m}\cdot i+\frac{m-1}{n}\left\lceil i\cdot\frac{n}{m}\right\rceil\geq\frac{5}{7}\left(1+i+\left\lceil i\cdot\frac{n}{m}\right\rceil\right)$$

which simplifies to

$$\frac{2}{7}+\left(\frac{n}{m}-\frac{5}{7}\right)i\geq\left(\frac{5}{7}-\frac{m-1}{n}\right)\left\lceil i\cdot\frac{n}{m}\right\rceil.$$

We can safely assume that $\frac{5}{7}-\frac{m-1}{n}\geq 0$, because otherwise the right-hand side would be negative, while the left-hand side remains positive. Since $\lceil x\rceil\leq x+1$, the following equality implies (c1)

$$\frac{2}{7}+\left(\frac{n}{m}-\frac{5}{7}\right)i\geq\left(\frac{5}{7}-\frac{m-1}{n}\right)\left(i\cdot\frac{n}{m}+1\right)$$

And simplifies to

$$\left(\frac{2}{7}\frac{n}{m}+\frac{m-1}{m}-\frac{5}{7}\right)i+\frac{m-1}{n}\geq\frac{3}{7}.$$

We can observe that $\frac{2}{7}\frac{n}{m}+\frac{m-1}{m}-\frac{5}{7}\geq\frac{2}{7}+\frac{1}{2}-\frac{5}{7}>0$, so that the left-hand side of this inequality is minimized with $i=1$. It is thus enough to check that

$$\frac{2}{7}\frac{n}{m}+\frac{m-1}{m}+\frac{m-1}{n}\geq\frac{8}{7}.$$

This is what we do here:

$$\frac{2}{7}\frac{n}{m}+\frac{m-1}{m}+\frac{m-1}{n}\geq\frac{2}{7}\frac{n}{m}+\frac{1}{2}+\frac{m-1}{m}\frac{m}{n}\geq$$

$$\geq\frac{1}{2}+\frac{2}{7}\frac{n}{m}+\frac{1}{2}\frac{m}{n}\geq\frac{1}{2}+2\cdot\sqrt{\frac{2}{7}\times\frac{1}{2}}\geq\frac{8}{7}.$$

Let us now check condition (c2). Here, we denote $|\pi|_\square$ by $i$ and $|\pi'|_\square$ by $j$ ($0\leq j\leq i<m\leq n$) (We only have to consider the longest $\pi'$). Condition (c2) becomes $\forall 0\leq j\leq i<m\leq n$

$$1+\left\lceil i\frac{n}{m}\right\rceil\frac{m-1}{n}+j\frac{n}{m}-\frac{5}{7}\left(1+\left\lceil j\frac{n}{m}\right\rceil+i\right)\geq 0.$$

Simplifying, and substituting $\lceil x\rceil$ by $x$ we get

$$\frac{2}{7}+i\cdot\frac{m-1}{m}+j\frac{n}{m}-\frac{5}{7}\left\lceil j\frac{n}{m}\right\rceil-\frac{5}{7}i\geq 0.$$

First, we solve the case when $j=0$

$$\frac{2}{7}+i\left(\frac{m-1}{m}-\frac{5}{7}\right)\geq 0$$

- $m=2$

$$\frac{2}{7}+i\left(\frac{1}{2}-\frac{5}{7}\right)\geq 0$$

$$i\leq\frac{4}{3}$$

that holds true since $i<m=2$

- $m=3$

$$\frac{2}{7}+i\left(\frac{2}{3}-\frac{5}{7}\right)\geq 0$$

$$i\leq 6$$

that holds true since $i<m=3$

- $m>3$

$$\frac{m-1}{m}\geq\frac{5}{7}.$$

Which solves case $j=0$.

Now we can safely assume $j\geq 1$. Using $\lceil\frac{a}{b}\rceil\leq\frac{a+b-1}{b}$, it is sufficient to prove that

$$\frac{2}{7}+i\cdot\frac{m-1}{m}+j\frac{n}{m}-\frac{5}{7}\left(j\frac{n}{m}+\frac{m-1}{m}\right)-\frac{5}{7}i\geq 0,$$

which simplifies to

$$\frac{2}{7}i+\frac{2}{7}j\frac{n}{m}+\frac{5}{7}\frac{1}{m}\geq i\frac{1}{m}+\frac{3}{7}.$$

Obviously, the left-hand side is minimized for $j=1$:

$$\frac{2}{7}i+\frac{2}{7}\frac{n}{m}+\frac{5}{7}\frac{1}{m}\geq i\frac{1}{m}+\frac{3}{7}.$$

We consider three cases:

- $m=2$ (then, $i=1$ must hold):

$$\frac{2}{7}+\frac{2}{7}\frac{n}{2}+\frac{5}{7}\times\frac{1}{2}\geq\frac{1}{2}+\frac{3}{7}$$

which is equivalent to $n\geq 2$ (that holds).

- $m=3$ (then, $1\leq i<3$ must hold)

$$\frac{2}{7}\frac{m}{3}+\frac{5}{7}\times\frac{1}{3}\geq\frac{1}{21}i+\frac{3}{7}$$

which is equivalent to $2n\geq i+4$ (that holds).

- $m\geq 4$

$$\left(\frac{2}{7}-\frac{1}{m}\right)i+\frac{2}{7}\frac{n}{m}+\frac{5}{7}\frac{1}{m}\geq\frac{3}{7}$$

right-hand side minimizes for $i=1$

$$\frac{2}{7}-\frac{1}{m}+\frac{2}{7}\frac{n}{m}+\frac{5}{7}\frac{1}{m}\geq\frac{3}{7}$$

which simplifies to $2n\geq 2+m$, that also holds.

■

### C. Proof of Theorem 6.3

*Proof (Theorem 6.3):* For a given $\alpha=\frac{p}{q}<1$, ($p$ and $q$ have integer values), and for any $k$, let us consider the instance $I(\alpha,k)$ such that:

- $b_0=1$;
- $n=kq$ open nodes have bandwidth $\alpha$; and
- $m=kp$ guarded nodes have bandwidth $\frac{1}{\alpha}$.

The first observation is that for all $\alpha$ and $k$, Lemma 5.1 implies that the optimal throughput $T^*$ is equal to 1.

For the second observation, let $S$ be any acyclic solution to $I(\alpha,k)$ and $x$ be the number of open nodes before the second guarded node in $S$. In other words, $S$ starts with a prefix $\pi=\bigcirc^u\square\bigcirc^v\square$ with $u+v=x$. The throughput $T$ achievable by $S$ is bounded by two constraints

- the source and the first $x$ open nodes should be able to feed the first two guarded nodes, ie. $\alpha x+1\geq 2T, and$
- the bandwidth of the source and of the first $x+1$ nodes should be enough to feed the $x+2$ nodes, ie. $\alpha x+\frac{1}{\alpha}+1\geq(x+2)T$.

Hence $T\leq\frac{\alpha x+1}{2}=f_\alpha(x)$ and $T\leq\frac{\alpha x+\frac{1}{\alpha}+1}{x+2}=g_\alpha(x)$. Since any optimal acyclic scheme must satisfy

these two constraints for some $x$, we have $T^*_{ac} \leq \max_{x\in\mathbb{N}} \min(f_\alpha(x), g_\alpha(x))$.

Observe now that the function $f_\alpha$ is increasing, and $g_\alpha$ is decreasing (since $\alpha < 1$), and that they coincide (with value 1) for $x = \frac{1}{\alpha}$. The minimum is thus achieved by $f_\alpha$ for $x < 1/\alpha$, and by $g_\alpha$ for $x > 1/\alpha$, and this minimum is maximized for $x = 1/\alpha$. However, $\frac{1}{\alpha}$ is not necessarily an integer, so the maximal value is achieved for $x = \left\lfloor \frac{1}{\alpha} \right\rfloor$ or $x = \left\lceil \frac{1}{\alpha} \right\rceil$:

$$T^*_{ac} \leq \max\left( f_\alpha\left(\left\lfloor \frac{1}{\alpha} \right\rfloor\right), g_\alpha\left(\left\lceil \frac{1}{\alpha} \right\rceil\right)\right).$$

If $\alpha = \frac{\sqrt{41}-3}{8}$, simple computations show that $\left\lfloor \frac{1}{\alpha} \right\rfloor = 2$, $\left\lceil \frac{1}{\alpha} \right\rceil = 3$, and $f_\alpha(2) = g_\alpha(3) = \frac{\sqrt{41}+1}{8}$. Since this value of $\alpha$ can be approximated arbitrarily close with a rational number, and since the expressions $f_\alpha(2)$ and $g_\alpha(3)$ are continuous in $\alpha$, we get the claimed result. ■ ■

## XII. Average Case

In addition to this worst-case analysis, we also analyze the average ratio between acyclic and cyclic throughput of randomly generated instances. In order to explore the performance of our algorithms in different heterogeneity conditions, we consider several probability distributions for the bandwidths of the nodes

1) an uniform distribution between 1 and 100 (**Unif100**);
2) power-law (Pareto) distributions with average value 100 and standard deviation 100 (**Power1**) or 1000 (**Power2**);
3) log-normal distributions with average value 100 and standard deviation 100 (**LN1**) and 1000 (**LN2**);
4) a uniform sampling from outgoing bandwidth values that were computed from measurements performed on the PlanetLab platform [14] (**PLab**).

In each case, each node is independently chosen to be an open node with probability $p$ (and a guarded with probability $(1-p)$). In order to concentrate on difficult instances, the bandwidth of the source node is chosen equal to the optimal cyclic throughput – what ensures that the source is not a strong limiting bottleneck, and that it is also not sufficient by itself to feed all nodes. The results are shown on Figure 19, for different numbers of nodes and different values of $p$. For each set of parameters, 1000 random instances were generated, and the figure shows average values (connected by black lines) and boxplots with median, quantiles, and confidence intervals at 5% (the black dots are outliers, outside these confidence intervals).

The first conclusion of these simulations is that the average behavior of acyclic solutions is very close to the optimal cyclic throughput, and that this is true in a wide variety of scenarios. Furthermore, the results are very stable. We can note that more open nodes and moderate heterogeneity (with the **Power1** and **Power2** distributions) make the problem slightly more difficult for small size instances. Overall, however, we can see that even in these cases, producing low degree solutions comes at very little cost (at most 5%) with respect to the achievable throughput.

The second conclusion is related to the acyclic throughput obtained considering only the best solution among those encoded by words $\omega_1$ and $\omega_2$ (blue lines on Figure 19). In all cases, these solutions are almost as competitive as the best acyclic ones and for all large instances they are as competitive. From a practical point of view these simpler schemes are of interest since they are easier to build in a distributed context once nodes have been ordered according their bandwidth. For comparison, the average throughput obtained by the word (either $\omega_1$ or $\omega_2$) used in the case analysis of the proof of Theorem 6.2 is shown by the red lines on Figure 19. We can see that there is a significant gap for smaller instances, hence it can be actually worthwhile to compute the best throughput among both words.

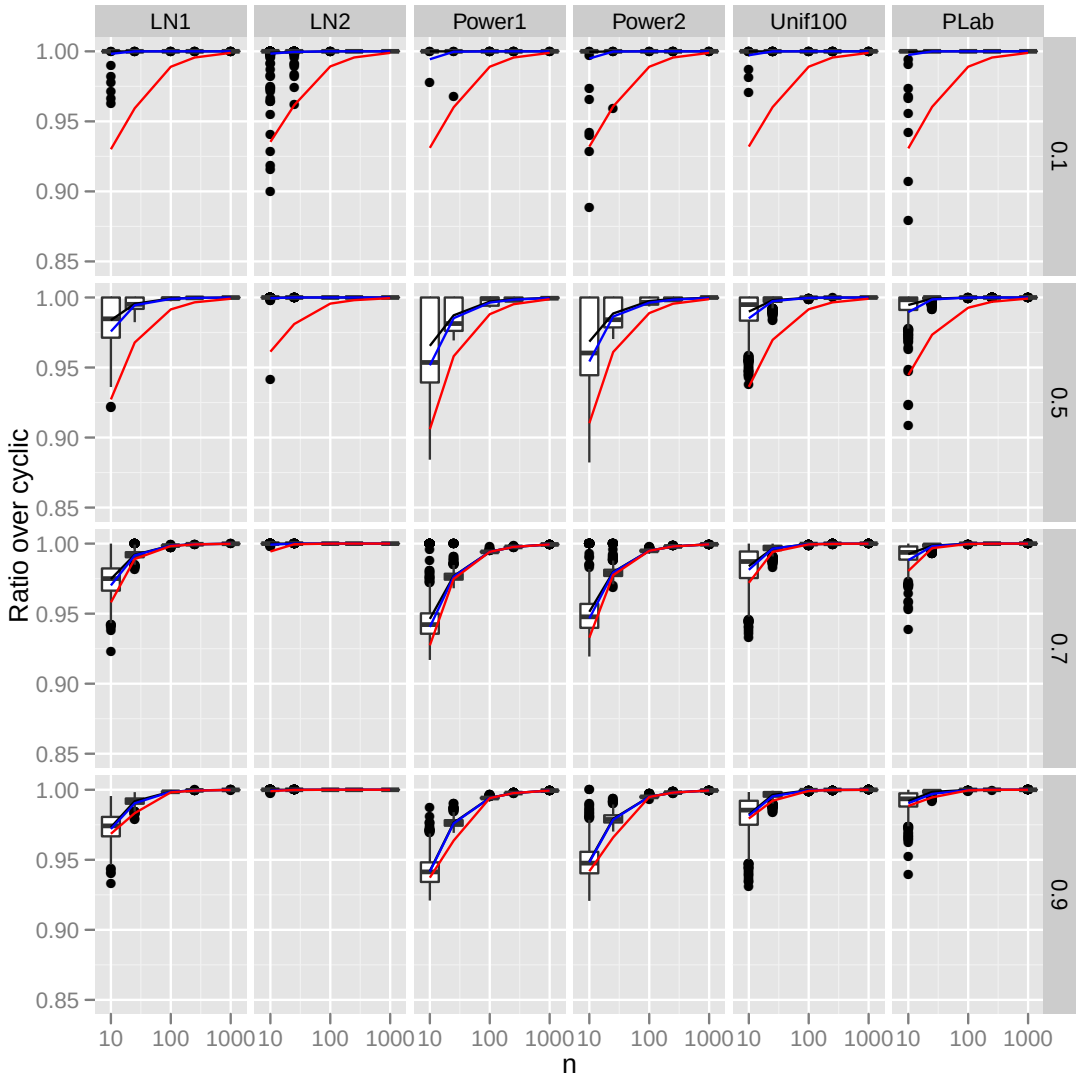


Figure 19. Throughput of acyclic solutions on randomly generated instances, normalized by the optimal cyclic throughput. Black boxplots show the optimal acyclic throughput, blue lines show the average throughput of the best solution among $\omega_1$ and $\omega_2$, red lines show the throughput of the solution (either $\omega_1$ or $\omega_2$) used in the proof of Theorem 6.2.