A Distributed Algorithm for Bandwidth Allocation in Stable Ad Hoc Networks

Claude Chaudet, Isabelle Guérin, Janez Zerovnik

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A Distributed Algorithm for Bandwidth Allocation in Stable Ad Hoc Networks

Claude Chaudet — Isabelle Guérin Lassous — Janez Žerovnik

N° 4827 – version 2

version initiale mai 2003 – version révisée octobre 2003

THÈME 1

ISSN 0249-6399

A Distributed Algorithm for Bandwidth Allocation in Stable Ad Hoc Networks

Claude Chaudet , Isabelle Guérin Lassous , Janez Žerovnik

Thème 1 — Réseaux et systèmes
Projet Ares

Rapport de recherche n° 4827 – version 2*— version initiale mai 2003 — version révisée octobre 2003 21 pages

Abstract: We propose a distributed algorithm for allocating bandwidth in stable ad hoc networks. After having discussed the problem of bandwidth allocation in such networks, we define a sequence of feasible solutions to this problem. This sequence has the property to be an increasing sequence in terms of overall used bandwidth. After a theoretical analysis of the sequence, we design a distributed algorithm based on this sequence. We test our algorithm by simulations on different topologies like chains, rings, meshes and geometric random graphs. We compare our solutions with the optimal solution in terms of global bandwidth allocation that presents the smallest standard deviation and with the the fairest solution regarding to max-min fairness. The simulations show that the global used bandwidth is less than 25% from optimality in the worst case and the standard deviation is the smallest of the three tested solutions.

Key-words: bandwidth allocation, ad hoc networks, distributed algorithm, fairness

* In this version of the report, the theoretical description of the problem has been clarified and the results have been recomputed and are analyzed more accurately. A comparison with the max-min most fair solution has been added.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38330 Montbonnot-St-Martin (France)
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Un algorithme distribué d'allocation de bande passante pour les réseaux ad hoc stables

Résumé : Dans ce papier nous proposons un algorithme distribué permettant d'allouer la bande passante dans des réseaux ad hoc stables. Après une discussion sur le problème de l'allocation de bande passante dans ce type de réseaux, nous définissons une suite de solutions à ce problème et montrons que cette suite conduit à une allocation de bande passante globale croissante. À partir de cette analyse théorique, nous dérivons un algorithme distribué et le testons sur différentes topologies (chaînes, anneaux, treillis et graphes géométriques aléatoires). Nous comparons notre solution à la solution optimale en terme de bande passante totale allouée qui présente le plus petit écart type entre les différents mobiles, ainsi qu'avec la solution la plus équitable au sens de l'équité max-min. Les simulations que nous avons effectuées montrent que la bande passante allouée représente au moins 75 % de la valeur optimale dans le pire cas et que, dans tous les cas notre solution est celle qui présente le plus faible écart type.

Mots-clés : allocation de bande passante, réseaux ad hoc, algorithme distribué, équité

Introduction

Today, most of the existing wireless radio networks (GSM, WiFi) are infrastructure based networks. A fixed base station manages the transmissions in a certain geographic zone corresponding to its transmission range. Mobiles in this area need to be connected directly to this base station in order to communicate.

Mobile ad hoc networks are an evolution of these wireless networks in which no fixed infrastructure is needed. Mobiles communicate directly between each other, without the need for a base station. To enable these communications, any mobile should be able to perform routing for the others. The MANET (Mobile Ad hoc NETworks) working group has been created at the IETF in order to standardize a routing protocol for ad hoc networks. This working group has arrived in the final phase of its researches and is about to decide what the best effort routing protocol for these networks will be.

In these networks, the radio medium is shared by all the mobiles. Many techniques are available to manage multiple access to the medium, from centralized or semi-centralized ones (time slot, frequency or spreading code allocation for example) to totally distributed ones (CSMA type for instance). Due to the infrastructure-less nature of ad hoc networks, the distributed medium access protocols seem more suited.

As most of the available wireless interface cards implement the IEEE 802.11 [6] standard, most of the actual ad hoc networks rely on it. This standard provides a totally distributed mode for the medium access part based on a CSMA / CA mechanism: the mobiles communicate on the same frequency and as long as one mobile emits, it prevents all of its neighbors from transmitting. Otherwise if two nearby mobiles emit simultaneously on the same frequency, the two signals interfere and there is a high probability that none of them results in a successful transmission. This means that as long as there is no congestion, mobiles can freely access the medium (in accordance with the 802.11 MAC protocol) and use whatever share of the resources they need. When the medium capacity is exceeded, the protocol behavior and the subsequent medium share is unpredictable.

Most of the proposed protocols for bandwidth management in ad hoc networks have an *a posteriori* approach, i.e. the bandwidth is managed only when congestion points appear. Protocols like INSIGNIA [8], SWAN [1] or HMP [9] do use such an approach. They provide notification, degradation or/and re-routing in order to react to the appearance of congestion. These approaches usually need additional communication between mobiles when the network already suffers from congestion. Therefore, returning to a stable situation can take some time. An alternative would be to manage the bandwidth *a priori*. If each mobile controls the use of its bandwidth rather than trying to use the whole radio medium, it will be able to prevent most of the congestion situations from appearing. It is difficult in such networks to avoid any congestion due to mobility: the control applied by one mobile in one configuration may not be adapted anymore in another configuration obtained by its mobility and may lead to the appearance of congestion. But an *a priori* solution may prevent from most of the congestion and may give efficient results in terms of bandwidth management as long as the network is quite stable with a low mobility.

Under such networks, mechanisms like bandwidth reservation are essentially provided for constrained traffic, for example real-time flows. If the admission control is well suited to the characteristics of the network, reservations can prevent congestion from appearing. Best effort traffic is usually not limited by any mechanism and can thus easily overlap on the privileged traffics' bandwidth share, making the guarantees fragile. One possible solution to this problem is to allocate a constant amount of bandwidth for best effort traffic but this does neither take into account the resources required for the privileged traffic nor the topology of the network. Such solution often leads to a sub-optimal use of the network resources. An alternative is to allocate bandwidth to best effort traffic according to the properties of the network, i.e. the topology and the bandwidth available to each mobile. This assignment is done so that no saturation appears on any mobile. Finding such a solution while maximizing at the same time the overall used bandwidth in the network is equivalent to a fractional packing problem. Algorithms solving this problem are essentially sequential and difficult to adapt to a distributed setting. Moreover, such solutions maximize the total used bandwidth without guaranteeing any fairness among the mobiles. Not providing a minimum amount of bandwidth for each mobile in the network may lead to serious imbalance and to a bad use of the network.

In this article, we propose a distributed algorithm to allocate bandwidth to each mobile according to the topology of the network and the available bandwidth on each mobile for stable ad hoc networks. The algorithm guarantees a non null minimum bandwidth to each mobile. With this algorithm, each mobile computes the bandwidth it can use in order to avoid saturating its capacity or its neighbors'. With such an algorithm, congestion is less likely to appear in the network.

In Sect. 1, we give the used model for ad hoc networks and the bandwidth allocation problem. A simple bandwidth allocation is given in Sect. 2. From this allocation, we design a sequence of feasible allocation in Sect. 3. Each term of the sequence is a new bandwidth allocation that is more efficient in terms of global used bandwidth than the previous terms in the sequence. Different properties of the sequence are described in this section. From this sequence, we design a distributed algorithm that allocates bandwidth to each mobile of the network in Sect. 4. At each step of the algorithm, each mobile needs only to know the minimum remaining bandwidth and the maximum degree in its neighborhood. In Sect. 4.3, we present the various results obtained on different network configurations like chains, rings, meshes and geometric random graphs. This work is on going and we evaluate in this article the quality of our allocation in terms of overall used bandwidth and of fairness among the mobiles. The obtained results should allow us to decide if our algorithm can be converted in an efficient bandwidth allocation protocol for ad hoc networks. In the conclusion, we discuss the points to solve for the protocol version and how to integrate mobility in our solution.

1 The Model and the Problem

We model our ad hoc network by a vertex weighted graph $G(V, E, b)$ where:

- $V = V(G)$ is the set of vertices of the graph. One vertex in the graph represents one mobile in the network;
- $E = E(G)$ is the set of edges of the graph. There is an edge between two vertices whenever the two corresponding mobiles are able to communicate, i.e. are in each other's transmission range;
- b is a function which assigns positive real numbers to vertices, representing the capacity of the medium around each mobile.

A congestion appears whenever the capacity of the medium is exceeded in a certain region of the network. If we suppose that mobiles only share the medium with their direct neighbors, ensuring that for each mobile in the network, the sum of its used bandwidth and of the bandwidth used by its neighbors does not exceed this mobile's capacity will ensure that there is no congestion at all in the network. Formally, if, for each mobile v in the network, we note $N[v] = \{v\} \cup \{u \mid uv \in E\}$ its closed neighborhood and $x(v)$ the amount of bandwidth that v can use, the problem can be expressed as:

$$\forall v \in V, \sum_{u \in N[v]} x(u) \leq b(v)\,. \tag{1}$$

Maximizing at the same time the overall use of the network is equivalent to the following problem:

$$\max \sum_{v \in V} x(v) \quad s.t. \quad \forall v \in V, \sum_{u \in N[v]} x(u) \leq b(v)\,.$$

This linear problem is known as a fractional packing problem. The problem can be solved by usual linear programming algorithms, and there are also faster approximation algorithms known (see [10] for details and for further references). These algorithms are sequential. A distributed algorithm for linear programming that obtains a $(1+\varepsilon)$-approximation in polylogarithmic number of communication rounds is given in [2]. Solutions to the fractional packing problem maximize the total bandwidth used in the network and give no guarantee on the minimum bandwidth for each mobile (i.e. the bandwidth may be null on some mobiles). This may have an impact on the good running of the network.

We propose a distributed algorithm that computes a set of values $\{x(v)\}_{v \in V}$ solution to the constraints (1). The algorithm is based on the topology of the network and the available bandwidth of each mobile. Before presenting the algorithm, we define the notation and formally evaluate the features of the solution in Sect. 2 and Sect. 3.

2 The Basic Lemma

We will use the following notations:

- $N[v]$ is the closed neighborhood of the vertex v,

- $N(v)$ is the open neighborhood of the vertex v ($N(v) = N[v] \setminus \{v\}$),
- $d(v)$ is the degree of vertex v,
- $\mathbf{x} = (x(v), v \in V)$ is the vector of values we are trying to compute, i.e. the solution of (1), i.e. from a networking point of view, the allocated bandwidth to the mobiles,
- $\mathbf{b} = (b(v), v \in V(G))$ is the vector of the "capacities" of the vertices, i.e. from a networking point of view, the available bandwidth for each mobile,
- $\Delta_1(v) = \max_{u \in N[v]} d(u)$, i.e. the maximum degree over the closed neighborhood of v (v included),
- $b_1(v) = \min_{u \in N[v]} b(u)$, i.e. from a networking point of view, the minimum bandwidth over the closed neighborhood of v (v included).

Lemma 2.1 *If for every node v in the graph, $x(v) = \frac{b_1(v)}{\Delta_1(v)+1}$, then $\mathbf{x} = (x(v), v \in V)$ is suitable to the constraints (1).*

Proof Recall from definitions of Δ_1 and b_1 that $u \in N[v]$ implies that $d(v) \leq \Delta_1(u)$ and $b(v) \geq b_1(u)$. Therefore,

$$\begin{aligned} \sum_{u \in N[v]} x(u) &\leq \sum_{u \in N[v]} \frac{b_1(u)}{\Delta_1(u)+1} \leq \sum_{u \in N[v]} \frac{b(v)}{d(v)+1} \leq \frac{d(v)+1}{d(v)+1} b(v) \\ &\leq b(v)\,. \end{aligned}$$

∎

It means that $\mathbf{x}$ can be used as an initial solution to the constraints (1) and is a feasible bandwidth allocation.

3 A Sequence of Feasible Vectors

In Sect. 2, we have found a vector of values that respects the constraints defined by (1). In this section, we are going to show that if we iterate the process using the remaining bandwidth at each mobile after having computed $\mathbf{x}$, then we still have a solution to the problem and we increase the overall used bandwidth in the network.

3.1 Sequence Definition

We will consider the following sequences of vectors:

- $x^{(i)}(v)$ represents the allocated resources amount for the node v at the i^{th} step of the sequence,

- $e^{(i)}(v)$ represents the remaining capacity of the node v at the i^{th} step considering what its neighbors have taken at this step.

These values are initialized as follows:

$$x^{(0)}(v) = \frac{b_1(v)}{\Delta_1(v)+1}\,.$$

$$e^{(0)}(v) = b(v) - \sum_{u\in N[v]} x^{(0)}(u)\,.$$

Then the passage from step i to step $i+1$ is done on the following way:

$$x^{(i+1)}(v) = x^{(i)}(v) + \frac{1}{\Delta_1(v)+1}.\min_{u\in N[v]} e^{(i)}(u)\,.$$

$$e^{(i+1)}(v) = b(v) - \sum_{u\in N[v]} x^{(i+1)}(u)\,.$$

3.2 Sequence Properties

Lemma 3.1 *All the terms of this sequence respect the constraints defined by (1), i.e.*

$$\forall v \in V, \forall i \in \mathbb{N},\ \sum_{u\in N[v]} x^{(i)}(u) \le b(v)\,.$$

Proof We can write, $\forall v \in V, \forall i \in \mathbb{N}$,

$$\forall u \in N[v],\ \min_{w\in N[u]} e^{(i)}(w) \le e^{(i)}(v)\,.$$

$$\Rightarrow \sum_{u\in N[v]} \left(\min_{w\in N[u]} e^{(i)}(w)\right) \le |N[v]| \times e^{(i)}(v)\,.$$

As $|N[v]| > 0$, we can write:

$$\frac{\sum_{u\in N[v]} \left(\min_{w\in N[u]} e^{(i)}(w)\right)}{|N[v]|} \le e^{(i)}(v)\,.$$

As $\forall u \in V, 1+\Delta_1(u) \ge 1+d(u)$ and $1+d(u) = |N[u]|$, then

$$\sum_{u\in N[v]} \min_{w\in N[u]} e^{(i)}(w) \times \frac{1}{\Delta_1(u)+1} \le e^{(i)}(v)\,.$$

Therefore

$$\sum_{u\in N[v]} (x^{(i+1)}(u) - x^{(i)}(u)) \le b(v) - \sum_{u\in N[v]} x^{(i)}(u)\,.$$

$$\Rightarrow \sum_{u \in N[v]} x^{(i+1)}(u) \leq b(v)\,.$$

We have proved that $\forall v \in V, \forall i \in \mathbb{N}^*, \sum_{u \in N[v]} x^{(i)}(u) \leq b(v)\,.$

Thanks to Lemma 2.1, we know that

$$\forall v \in V, \sum_{u \in N[v]} x^{(0)}(u) \leq b(v\,.)$$

Therefore,

$$\forall v \in V, \forall i \in \mathbb{N}, \sum_{u \in N[v]} x^{(i)}(u) \leq b(v)\,.$$

∎

It means that each element of the sequence is a solution to the constraints (1) and is a feasible bandwidth allocation.

Lemma 3.2 *If the vector* **b** *contains no zero then every allocated value is non-zero, i.e.* $\forall v \in V, \forall i \in \mathbb{N}, x^{(i)}(v) > 0\,.$

Proof Straightforward with the definition of $x^{(0)}(v)$ and Lemma 3.1. ∎

It means that each term of the sequence corresponds to a bandwidth allocation where no mobile has a bandwidth equal to 0.

Lemma 3.3 *The sequence* $\left(x^{(i)}(v)\right)_{i \in \mathbb{N}}$ *is convergent.*

Proof We can write, using the definition of $e^{(i)}(v)$ and Lemma 3.1, that $\forall v \in V, \forall i \in \mathbb{N}, e^{(i)}(v) \geq 0$. Therefore, the sequence $\left(x^{(i)}(v)\right)_{i \in \mathbb{N}}$ is monotone increasing. As a monotone increasing and bounded sequence always converges, and as $\left(x^{(i)}(v)\right)_{i \in \mathbb{N}}$ is bounded by $b(v)$, $\left(x^{(i)}(v)\right)_{i \in \mathbb{N}}$ converges. ∎

It means that each term of the sequence has an overall bandwidth greater than the previous terms in the sequence and that the sequence tends to a solution that has the maximum global bandwidth within this sequence.

Lemma 3.4 *A node v will reach a null remaining bandwidth at step i (i.e. $e^{(i)}(v) = 0$) if and only if this node has the minimum remaining bandwidth and the maximum degree among its neighbors at step $i-1$.*

Proof Assume that a node v has a non null remaining bandwidth at step $i-1$ (i.e. $e^{(i-1)}(v) > 0$) and does not have the maximum degree among its neighbors (i.e. $d(v) < \Delta_1(v)$ and $\exists z \in N(v) | d(z) = \Delta_1(v)$), by definition :

$$\begin{aligned} e^{(i)}(v) &= e^{(i-1)}(v) - \sum_{u \in N[v]} \frac{\min_{w \in N[u]} e^{(i-1)}(w)}{\Delta_1(u)+1} \\ &= e^{(i-1)}(v) - \left(\sum_{u \in N[v] \setminus \{z\}} \frac{\min_{w \in N[u]} e^{(i-1)}(w)}{\Delta_1(u)+1} \right) - \frac{\min_{w \in N[z]} e^{(i-1)}(w)}{\Delta_1(z)+1} . \end{aligned}$$

As $\forall u \in N[v], d(v) \leq \Delta_1(u)$ and as $\forall w \in N[u], \min_{w \in N[u]} e^{(i-1)}(w) \leq e^{(i-1)}(v)$, then we have:

$$\frac{\min_{w \in N[u]} e^{(i-1)}(w)}{\Delta_1(u)+1} \leq \frac{e^{(i-1)}(v)}{d(v)+1} .$$

And as $\Delta_1(z) \geq d(z) > d(v)$:

$$\frac{\min_{w \in N[z]} e^{(i-1)}(w)}{\Delta_1(z)+1} > \frac{e^{(i-1)}(v)}{d(v)+1} .$$

Injecting these two expressions in the previous one, we obtain:

$$\begin{aligned} e^{(i)}(v) &> e^{(i-1)}(v) - d(v).\frac{e^{(i-1)}(v)}{d(v)+1} - \frac{e^{(i-1)}(v)}{d(v)+1} \\ &> 0 . \end{aligned}$$

Therefore, if a node has a non null bandwidth and does not have the maximum degree among its neighbors at a certain step, it will not reach its allocation limit at the next step. It is easy to show the same property for any node v that has a non null remaining bandwidth at a given step and has a minimum remaining bandwidth greater than one of its neighbors'. Therefore a node can reach its limit at a given step only if it had the minimum remaining bandwidth and the maximum degree among its neighbors at the previous step.

Proving that a node with minimum remaining bandwidth and maximum degree among its neighbors at some step reaches its limit at the next step is straightforward with the definition of $x^{(i)}$. ∎

This lemma characterizes the nodes that can reach a null remaining bandwidth. When they reach this state, their allocated bandwidth won't increase anymore in the next steps of the sequence and will remain constant. Note that the neighbors of such nodes will also reach the limit of their allocated bandwidth: they can't take more bandwidth in the next steps of the sequence otherwise they will saturate the capacity of the neighbor nodes that have a null remaining bandwidth.

3.3 Quality of the Solution

Lemma 3.5 *Every node has at least a neighbor whose free bandwidth converges towards 0:*

$$\forall v \in V, \exists u \in N[v] | \lim_{i \to +\infty} e^{(i)}(u) = 0 .$$

Proof As $\left(x^{(i)}(v)\right)_{i\in\mathbb{N}}$ converges, using the definition of the terms of the sequence: $x^{(i+1)}(v) = x^{(i)}(v)+\frac{1}{\Delta_1(v)+1}\min_{u\in N[v]} e^{(i)}(u)$, we can see that the number $x^{(i+1)}(v)-x^{(i)}(v) = \frac{1}{\Delta_1(v)+1}\min_{u\in N[v]} e^{(i)}(u)$ converges towards 0. Therefore, $\min_{u\in N[v]} e^{(i)}(u)$ converges towards 0. ∎

Lemma 3.6 *The sequence* $(x^{(i+1)}(v) - x^{(i)}(v))_{i\in\mathbb{N}}$ *is decreasing.*

Proof As the sequence $\left(x^{(i)}(v)\right)_{i\in\mathbb{N}}$ is increasing, then the sequence $\left(e^{(i)}(v)\right)_{i\in\mathbb{N}}$ is decreasing. Then:

$$\forall u \in N[v], e^{(i)}(u) \leq e^{(i-1)}(u)\,.$$

Therefore,

$$\exists z \in N[v] | \min_{u\in N[v]} e^{(i-1)}(u) = e^{(i-1)}(z) \geq e^{(i)}(z) \geq \min_{u\in N[v]} e^{(i)}(u)\,.$$

And

$$(x^{(i+1)}(v) - x^{(i)}(v)) \leq (x^{(i)}(v) - x^{(i-1)}(v))\,.$$

∎

From now on, we will note $X(v)$ the limit of the sequence $\left(x^{(i)}(v)\right)_{i\in\mathbb{N}}$, X the limit vector solution and $E(v)$ the remaining bandwidth at node v with the solution X.

Lemma 3.7 *X is Pareto-efficient.*

Proof We consider the order $\leq$ where $\leq$ is the natural order on $\mathbb{R}^N$, i.e. $x \leq y$ iff $x_i \leq y_i, \forall i \in [1, N]$. A solution is Pareto-efficient [5] if it is maximum in the sense of $\leq$.

Assume we have a vector S that respects the constraints (1) and so that $S > X$ according to the order previously defined. This means that there exists a node v so that $S(v) > X(v)$. According Lemma 3.5, there exists a node $u \in N[v]$ that has reached its bandwidth allocation limit. This means that $\sum_{w\in N[u]} X(w) = b(u)$. As $S(v) > X(v)$ and as S respects the constraints (1), necessarily there exists a node $z \in N[u] \setminus \{v\}$ such that $S(z) < X(z)$. Therefore we can not have $S > X$. ∎

3.4 Convergence Speed

Not much can be said on this algorithm convergence speed. The speed is highly dependent on the graph topology and on the bandwidths available at each node.

On one hand, for a regular graph with uniform weights ($\forall v \in V, b(v) = b$), the algorithm finds the optimal solution for the fractional packing problem at the first step (i.e. $\forall v \in V, x^{(0)}(v) = \frac{1}{1+\Delta}b$ and $e^{(0)}(v) = 0$).

On the other hand, we can find networks configurations that result in a sequence that converges with an infinite number of steps. For example, consider the 3-nodes chain on Fig. 1. Assume that $b(1) = b(3) = 2_1$ and $b(2) = 3$. After 1 step, the remaining bandwidth at each node is exactly divided by 3 (2/3 for nodes 1 and 3 and 1 for node 2) but the bandwidth keep the same ratios between each other. Therefore, all further steps will lead to

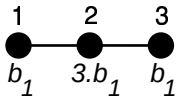

Figure 1: A three-nodes chain

a configuration in which the ratio between the bandwidths is the same as initially and the number of steps to converge is, in this configuration, infinite.

Now, if we consider the same configuration with $b(1) = b(3) = 2.b(2)$, the algorithm converges in a single step.

As we will show in Sect. 4.3, simulations give good results in the number of steps of our algorithm whatever the network may be.

4 A Distributed Algorithm for Bandwidth Allocation

4.1 Base algorithm

The following distributed algorithm Algorithm 1 is based on the sequence presented in Sect. 3. It computes an increasing sequence and each element of the sequence is a solution to the constraints (1). The algorithm locally computes in each vertex the sequence $\left(x^{(i)}(v)\right)_{i\in\mathbb{N}}$ stored in the variable X. At the end of the algorithm, X gives the bandwidth allocated at node v. The remaining bandwidth $\left(e^{(i)}(v)\right)_{i\in\mathbb{N}}$ is stored in each vertex in the variable E. The information that needs to be gathered for the computation consists, at each step, in the values $(x^{(i)}(u), e^{(i)}(u))_{i\in\mathbb{N}}$ for each neighbor and the degree of each neighbor. These informations need to be broadcasted in two steps as the calculation of one sequence requires the updated value of the other sequence.

4.2 Remarks on this Algorithm

- By Lemma 3.3, the algorithm clearly converges to a feasible solution X. Each term of the sequence $X^{(i)}$, represents a feasible solution for constraints (1) and gives the bandwidth that can be used by each mobile in the network. Each term $X^{(i)}$ gives an overall bandwidth greater than with the previous terms, but note that X is not always optimal in terms of global bandwidth as some examples presented in the next section will show.

- By Lemma 3.1, we know that the bandwidth allocated to the mobile with Algorithm 1 does not exceed the capacity of the network. This allocation is fair in the sense that no mobile overlaps on its neighbors resources amount. Moreover by Lemma 3.2 all mobiles do get some resources.

Algorithm 1: **: Bandwidth allocation** (at node v)

Input: the list of neighbors of v and $b(v)$ the available bandwidth at node v

Output: X the bandwidth allocated at node v

$E := b(v)$;

$X := 0$;

while *X is not constant* **do**

  send E and d to all neighbors;

  receive $E(u)$ and $d(u)$ from all neighbors u;

  $X := X + \frac{1}{\Delta_1(v)+1} \min E(u)$; (minimum over closed neighborhood)

  send X to all neighbors;

  receive $X(u)$ from all neighbors;

  $E := E - \sum X(u)$;

- By Lemma 3.5, we know that there is no "space left" in the resources once the algorithm is finished. That does not mean that all mobiles may use the total capacity of their wireless cards, but that means that each mobile has a neighbor in its closed neighborhood that has no space left. Thus, if a mobile wants to use more bandwidth than the one allocated then a congestion point will appear in its neighborhood because it will exceed the capacity of its neighbor that has no capacity remaining.

- By Lemma 3.6, we know that the difference between two consecutive values of X is decreasing. According to the quality of the solution we want to obtain, we can consider, at each step of the algorithm, that this difference becomes small enough to accept this solution and to not go further with the algorithm. Therefore, instead of achieving X constant, we can stop the algorithm when the difference between two consecutive values of X is smaller than a given threshold. We discuss the impact of this threshold in the next section.

- We also designed and tested a simple optimized version. In this second version, the nodes also transmit the minimum bandwidth in their neighborhood. If this value is 0, then neighbors will know this node will not be able to take anymore bandwidth. Then, they don't have to take it into account in the max degree calculation in the next step. They nevertheless still need to be considered when looking at the minimum bandwidth in the neighborhood. This optimized version may have a strong impact on the convergence with some configurations. For instance consider the configuration given in Fig. 2. Assume that the nodes have initially the same bandwidth. After one step of the algorithm, node 3 has no remaining bandwidth (i.e. equal to 0). It is not the case for node 4 because of Lemma 3.4. But this node can not take any bandwidth in the next round, otherwise it will exceed the capacity of node 3. Therefore only node 5 goes in the next round and gets more bandwidth. Moreover, with our initial algorithm node 5 will only take one third of its available bandwidth. So it will never fill its

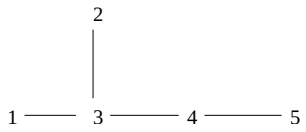

Figure 2: Impact of the optimized version

capacity and the algorithm will have an infinite number of steps. With the optimized version, node 4 will not be considered by node 5 for the max degree calculation and due to the configuration of the network node 5 will saturate its capacity or the one of node 4 in the next step. It means that the optimized algorithm converges in two steps with this example. In the next section, we compare the convergence times between the two versions of the algorithm.

4.3 Implementation Results of this Algorithm

Algorithm 1 has been implemented in C++. It has been tested on different kinds of configurations, like complete graphs, chains, rings, meshes and geometric random graphs. 50 runs of each test have been carried out on different sizes of the configurations and we give the average results.

To determine the quality of our algorithm, we compare our solution with a solution to the fractional packing problem and to the most fair solution respecting the constraints (1) regarding a max-min fairness criteria. A solution to the fractional packing problem may be computed by classical linear programming using the simplex algorithm. This results in a solution respecting the constraints (1) that maximizes the overall used bandwidth but that is not necessarily fair, as it may assign 0 bandwidth to some nodes. As the optimal solution to the linear programming maximization problem defined by (1) is not unique, we needed to find the "most fair" of these solutions. We chose to use standard deviation between the allocated bandwidths as the fairness criteria. To find the most fair optimal solution according to this criteria, we need to minimize the standard deviation between the allocated bandwidths subject to the constraints defined by (1) to which we add another constraint: the sum of all the bandwidths has to be equal to the objective function value computed by linear programming. This quadratic problem was solved using OOQP [7] and the results are presented below. As the time complexity of quadratic programming relatively large and the computation is sequential, the properties of the solutions can only be used as benchmarks for comparison and evaluation of the results of our algorithms.

We have also computed the most fair solution regarding to the max-min fairness criteria. This solution is often considered as the fairest allocation, as noted in [4, 3]. As a quick overview, in the max-min fairest solution to a problem, no bandwidth can be increased without decreasing another bandwidth that is already lower than the first one. Our solution is not the max-min fairest solution (it is easy to find a counterexample). We compare our solution with the max-min optimal solution.

For each configuration, we will discuss essentially three results: we compare the total bandwidth of the obtained solution with Algorithm 1 with the fractional packing optimal solution and with the max-min fairest solution; this comparison gives an idea on the quality of the solution in terms of global bandwidth use. We also compare the mean value and the standard deviation of the obtained solution with the two other solutions; these results give an indication on the fairness of the solution. Lastly, we compare the convergence times of the initial algorithm and the modified algorithm as described in Sect. 4.2. Each result has been obtained with different stopping thresholds (as discussed in Sect. 4.2). Three values of this threshold have been considered: 10%, 1% and 0.1% (it means that for a threshold of 1% for instance, the algorithm stops as soon as two consecutive values of X differ from less than 1%). Each node is given a random capacity (available bandwidth b) between 50 and 150. The results are gathered in the following paragraphs.

4.3.1 Complete Graphs (constant degree).

For a complete graph of n node, the algorithm always converges in one round, each node gets one n^{th} of the bandwidth and no capacity is left in the whole network. The solution is optimal and totally fair between the nodes.

4.3.2 Chains and Rings.

In chains, the degree is almost constant (2) except for the extreme nodes which only have one neighbor, whereas a ring is a graph in which each vertex is connected to exactly two neighbors.

Figure 3 compares the global bandwidth obtained with our algorithm with the one of an optimal solution and the one of a max-min solution. The left figure gives the results for the chains and the right figure gives the results for the rings. The number of nodes ranges from 2 to 100. We only give the results for two values of the stopping thresholds (10% and 1%), because the difference between the results with 1% and with 0.1% is extremely small. The results between the two configurations are very similar: the obtained global bandwidth is less than 10% under the optimal bandwidth; we have a better approximation with a stop threshold at 1% than at 10% and the difference is around 5%; the difference between the max-min solution and our solution is extremely small.

Figure 4 gives different statistics on the obtained solution with Algorithm 1 on the chains (left figure) and on the rings (right figure). We give our solution with a stop threshold at 1%. The results between the two configurations are very similar: of course, the same comparison can be done with the mean value as with the global bandwidth; the standard deviation is two times greater with the optimal solution than with the two other ones; the values allocated to the nodes with our solution (and the max-min one) range from around 17 to 37 in average, which is quite fair; the standard deviation of our solution is slightly smaller than the max-min one; the mean and the standard deviation are almost constant and are thus independent of the number of nodes. Note that the minimum values (not given in the figure) are the same with the two thresholds of our algorithm which means that some nodes

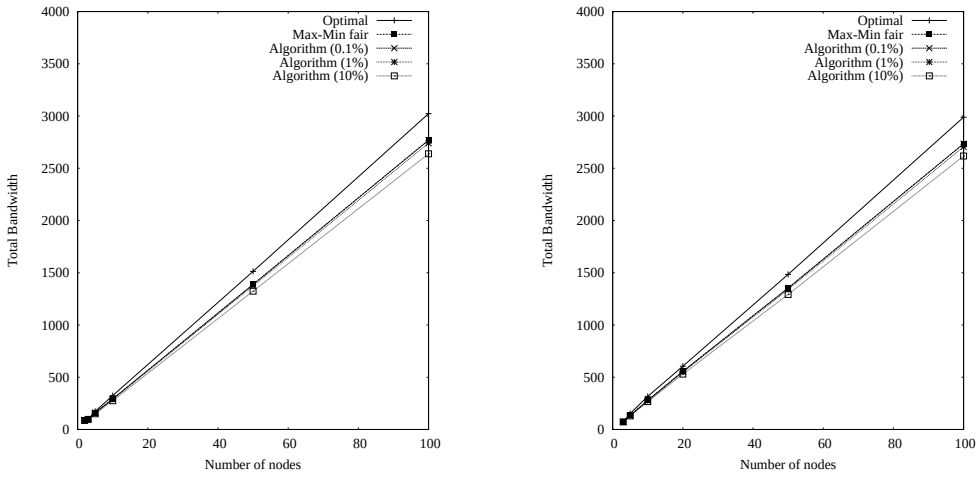

Figure 3: Global bandwidth on chains (left) and on rings (right)

have already no space left when two consecutive values of X in the algorithm differ from less than 10%.

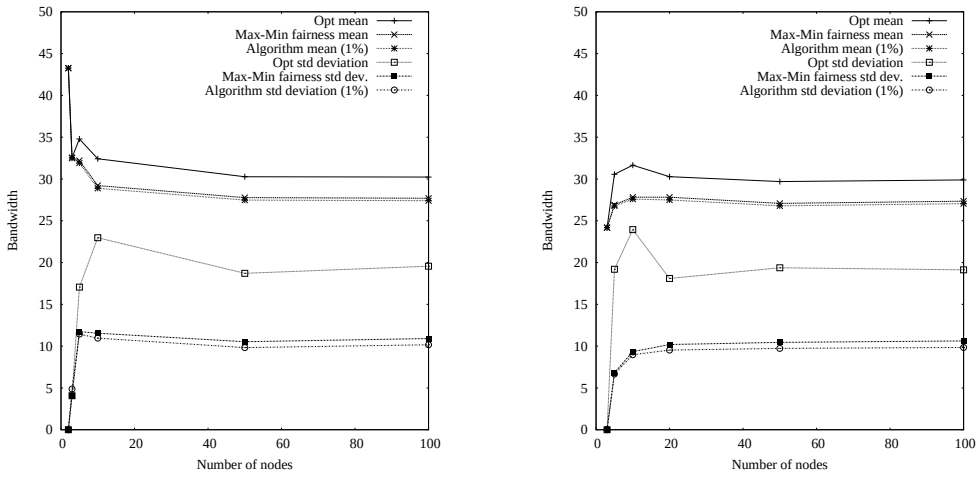

Figure 4: Statistics on the obtained solution on chains (left) and on rings (right)

Figure 5 gives the number of steps of convergence of Algorithm 1 on the chains (left figure) and on the rings (right figure). The results between the two configurations are very similar: the number of steps is independent on the number of nodes in the configuration,

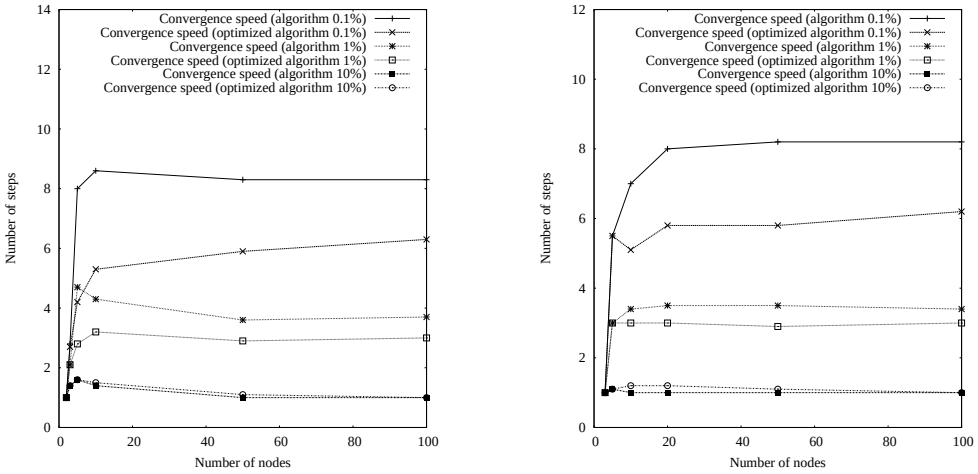

Figure 5: Number of steps of Algorithm 1 on chains (left) and on rings (right)

which shows that the problem has local properties; the maximum convergence speed is obtained with the initial algorithm with a threshold of 0.1% and is not high (8 steps); the modified version speeds up the convergence time and this speeding up increases with the refining of the threshold.

4.3.3 Meshes.

The size of the meshes ranges from 2×2 to 10×10. The approximate solution computed by our algorithm is around 15% smaller than the optimal one as it can be seen on Fig. 6 (left figure). As for chains and rings, our solution is almost equivalent to the max-min solution in terms of overall bandwidth. For the same reason as for the chains and the rings, we only give the results for a threshold of 10% and 1%. We also have a better approximation with a stop threshold of 1% than with 10% (difference of around 6%). The right figure of Fig. 6 shows different statistics on the obtained solution on meshes. The same remarks as for chains and rings can be done: the mean value is the greatest for the optimal solution and its standard deviation is more than two times greater than the one of our solution; the standard deviation of our solution is slightly smaller than the max-min one; the values allocated to the nodes with our algorithm range from around 9 to 25 in average, which is quite fair; the standard deviation is also independent of the size of the meshes. As for chains and rings, some nodes have already no space left when two consecutive values of X in the algorithm differ from less than 10%.

Figure 7 gives the number of steps taken by Algorithm 1 to converge. As for the chains and rings, the maximum convergence speed is obtained with the initial algorithm with a threshold of 0.1% and is around 12 steps; the number of steps increases with the refining of

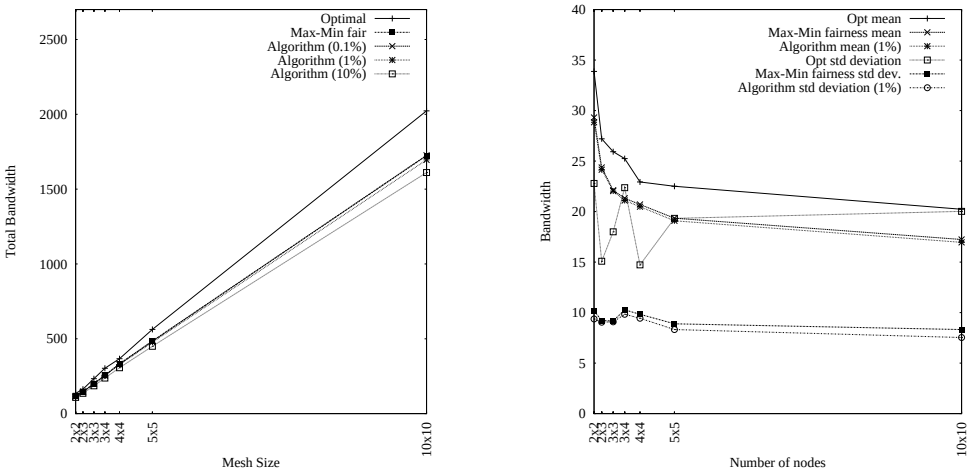

Figure 6: Global bandwidth and statistics on the obtained solution on meshes

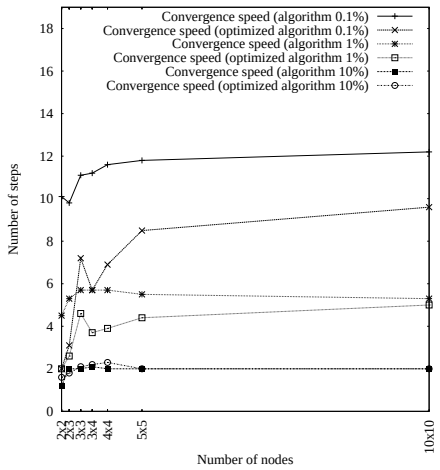

Figure 7: Number of steps of Algorithm 1 on meshes

the stop threshold and is independent of the size of the mesh; the modified version speeds up the convergence time and the speeding up increases with the refining of the stop threshold.

4.3.4 Geometric Random Graphs.

Our geometric random graphs are generated by considering 100 nodes put in a 1000×1000 square and by increasing the communication range of each node. The communication area ranges from 130 to 200 meters. When the communication range increases, the average degree of the nodes also increases.

As shown on Fig. 8 (left figure), the global bandwidth decreases with the communication range. This fact can be explained by the increase of the constraints of the system due to the increase of the average degree of the nodes and by the constant number of nodes. The obtained bandwidth with our algorithm is between 18% and 25% smaller than the optimal one. The overall bandwidth with the max-min solution is slightly greater than with our solution. The increase in global bandwidth is around 9% for 200 meters when the threshold switches from 1% to 0.1%. On the other hand, this increase is much smaller with sparse networks (it is around 5% with 130 meters). The statistics (right figure) show that our algorithm is also still fair with this configuration, because the values allocated to the nodes range from around 4 to 24 in average for a communication range of 130 meters and from around 1 to 11 in average for a communication range of 200 meters. Our algorithm is fairer on dense networks than on sparse ones. As for the other configurations, the standard deviation is around more than two times greater with optimal solution than with our solution. The difference between the max-min standard deviation and our's is more visible with these networks and is about 10%. As for the other tested configurations, some nodes have already no space left when two consecutive values of X in the algorithm differ from less than 10%.

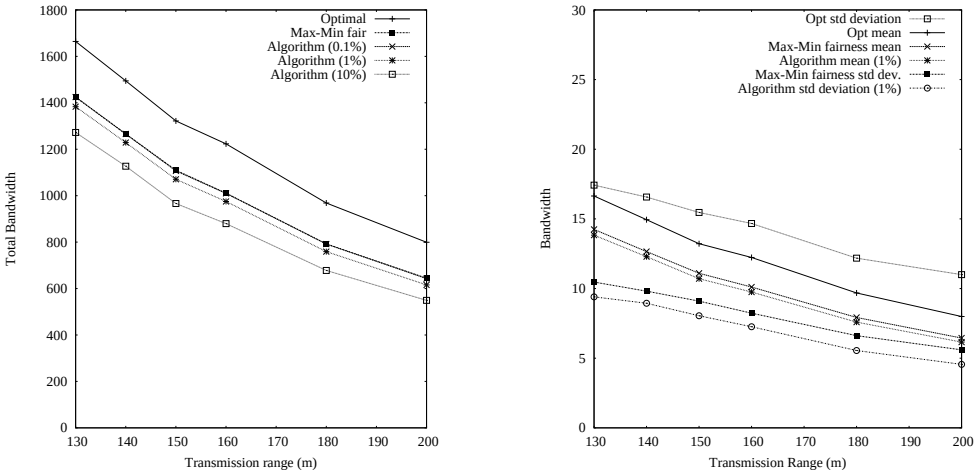

Figure 8: Global bandwidth and statistics on the obtained solution on geometric graphs

Figure 9 shows that the convergence speed depends on the degree of the nodes, especially when the threshold is 0.1%. The maximum convergence speed is obtained with the initial

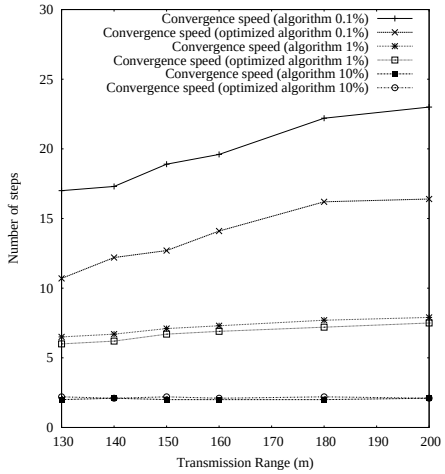

Figure 9: Number of steps of Algorithm 1 on geometric graphs

algorithm with a threshold of 0.1% and is around 22 steps for the worst situation with a communication range of 200 meters. In this configuration, the convergence is reduced to 16 steps with the modified version. As for the other configurations, the speeding up increases with the refining of the threshold.

By comparing the results for chains, rings and meshes and the results for geometric random graphs, we can say that the number of nodes in the networks has a limited impact on the quality of our solution (mean value, standard deviation and convergence) as long as the degrees remain constant. On the other hand, the degree is an important parameter and our algorithms perform better on dense graphs regarding fairness than on sparse ones and better on sparse graphs than on dense graphs regarding total bandwidth allocation and convergence speed.

Conclusion and Future Work

In this paper, we have presented and analyzed an algorithm for allocating bandwidth in stable ad hoc networks in order to minimize the appearance of congestion points. This algorithm is based on the computation of a sequence of solutions respecting a set of constraints defined by the radio medium capacity. The sequence is increasing and each solution of the sequence has an overall bandwidth greater than the previous terms of the sequence. The allocation computed by this algorithm is intended to be a good compromise between total bandwidth allocation and allocation fairness.

This algorithm has been implemented and simulation results obtained show that it converges towards a solution that is at the same time efficient regarding total bandwidth al-

location and quite fair. The solution is not an optimal for any of these two criteria but is quite near to optimal for both of them. The difference between the overall bandwidth of our solution and the one of an optimal one is less than 25% in the worst case. At the same time, the standard deviation of our solution is the smallest among the three tested solutions (a "most fair" optimal solution as described in Sect. 4.3, the max-min solution and ours'). The convergence speed can theoretically become very large but stopping near the solution instead of trying to reach it maintains the number of steps needed quite low. The proposed optimized version speeds up the convergence. The quality of the solution depends on the degrees of the mobiles in the network. At the same time, our algorithm is insensitive to the number of mobiles in the network as soon as the degrees remain constant.

The results achieved by this algorithm are promising and encourage us to convert it into a protocol in order to test its stability and to determine its influence on congestion points appearance.

Nevertheless, this conversion will not be straightforward. Mobility is also a key issue. Ad hoc networks are supposed to be mobile networks and the protocol will have to take into account topology changes. One solution is for each mobile to check the feasibility of the allocation considering appearing links regarding the constraints (1). If one link violates one of the constraints, the involved mobiles on the link may then share the minimum initial allocated bandwidth between these two nodes. If this solution quickly backtracks to a feasible solution, it may lead to an allocation less fair than the proposed solution of this article. Another alternative is that a mobile suffering an over-allocation implies its neighbors in the return to a stable solution. This kind of mechanism would probably lead to a fairer allocation but it could take some time before stabilizing. Our future work is to investigate such solutions.

References

[1] Gahng-Seop Ahn, Andrew T. Campbell, Andras Veres, and Li-Hsiang Sun. SWAN: Service differentiation in stateless wireless ad hoc networks. In *IEEE INFOCOM' 2002*, New York, USA, June 2002.

[2] Yair Bartal, Johna W. Byers, and Danny Raz. Global optimization using local information with applications to flow control. In *38th IEEE Symp. on Foundations of Computer Science*, pages 303–312, 1997.

[3] D. Bertsekas and R. Gallager. *Data Networks*. Prentice Hall, 1987.

[4] T. Bonald and L. Massoulié. Impact of Fairness on Internet Performance. In *Proceedings of SIGMETRICS*, Cambridge, MA, USA, June 2001.

[5] Allan M. Feldman. *Welfare Economics and Social Choice Theory*. Kluwer, Boston, 1980.

[6] IEEE Standard for Information Technology Telecommunications and Information Exchange between Systems. *Local and Metropolitan Area Network – Specific Requirements –Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. The Institute of Electrical and Electronics Engineers, 1997.

[7] Mike Gertz and Steve Wright. Object-oriented software for quadratic programming. Technical Report ANL/MCS-P891-1000, Argonne National Laboratory, Mathematics and Computer Science Division, 2001.

[8] Seoung Bum Lee, Gahng Seop Ahn, Xiaowei Zhang, and Andrew T. Campbell. Insignia: An ip-based quality of service framework for mobile ad hoc networks. *Journal on Parallel and Distributed Computing*, 60(4), 2000.

[9] Seoung-Bum Lee and Andrew T. Campbell. HMP: Hotspot mitigation Protocol for Mobile Ad Hoc Networks. In *11th IEEE/IFIP International Workshop on Quality of Service*, Monterey, Canada, June 2003.

[10] Serge A. Plotkin, David B. Schmoys, and Éva Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics Of Operations Research*, 20:257–301, 1995.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38330 Montbonnot-St-Martin (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399