Broadcast in Self-Organizing Multi-hop Wireless Networks

Nathalie Mitton, Anthony Busson, Eric Fleury

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Broadcast in Self-Organizing Multi-hop Wireless Networks

Nathalie Mitton - Anthony Busson - Eric Fleury

N° 5487

Fevrier 2005

Thème COM

ISSN 0249-6399 ISRN INRIA/RR--5487--FR+ENG

Broadcast in Self-Organizing Multi-hop Wireless Networks

Nathalie Mitton - Anthony Busson - Eric Fleury

Thème COM — Systèmes communicants
Projet ARES

Rapport de recherche n° 5487 — Fevrier 2005 — 37 pages

Abstract: Multi-hop wireless networks (MWN) consist of sets of mobile wireless nodes without the support of a pre-existing fixed infrastructure. Each host/node acts as a router and may arbitrary appear or vanish. This feature is a challenging issue for protocol design since protocols must adapt to frequent changes of network topologies. When dealing with sensor networks, the scalability also becomes a crucial aspect. In such large networks, we need not only to be able to route messages from any node to any other node but also to spread some information over the whole network. Till nowadays, it seems that these two properties have only been studied separately. In this report, we propose to use our existing cluster formation algorithm also to perform the broadcast operation.

Key-words: ad hoc, sensors, wireless, self-organization, broadcast, scalability

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38334 Montbonnot Saint Ismier (France)
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Etude de diffusion d'informations dans un réseau sans fils auto-organisé grande échelle

Résumé : Les réseaux multi-sauts sans fils (ou MWN *Multi-hop wireless networks*) consistent en des ensembles de stations mobiles communiquant uniquement par radio sans aucune infrastructure fixe. Chaque entité est entièrement mobile et peut apparaître ou disparaître indépendamment des autres. Afin que deux stations n'étant pas à portée radio l'une de l'autre puissent tout de même communiquer, chacune d'elles joue un rôle de routeur et un protocole de routage et/ou d'organisation est nécessaire. Dans de tels réseaux, il est également souvent nécessaire de propager une information à l'ensemble des nœuds. De nos jours, ces réseaux atteignent de grandes proportions et router et propager une information sur l'ensemble du réseau deviennent des opérations non triviales. Néanmoins, ces deux aspects (routage et diffusion) ne semblent pas avoir été étudiés conjointement jusqu'à maintenant. Nous nous proposons ici d'utiliser pour la diffusion une structure *clusterisée* du réseau, introduite pour le routage. De cette façon, nous ne créons qu'une structure mais pour deux utilisations.

Mots-clés : multi-sauts, ad hoc, senseurs, auto-organisation, robustesse, extensibilité, diffusion

1 Introduction

Multi-hop wireless networks consist of sets of mobile wireless nodes without the support of a pre-existing fixed infrastructure. Each host/node acts as a router and may arbitrary appear or vanish. This feature is a challenging issue for protocol design since protocols must adapt to frequent changes of network topologies. When dealing with sensor networks, the scalability becomes also a crucial aspect.

We need not only to be able to route messages from any node to any other node but also to spread some information over the whole network. In such large networks, flat routing protocols (reactive or proactive) or blind flooding are not really suitable. Indeed, such protocols become ineffective, because of bandwidth occupation, processing overhead (routing table computation), memory capacities and energy consumptions. Several solutions have been proposed for solving this scalability problem for each of these two functionalities. The most common one used for routing is to build a structure over the network to introduce a hierarchical routing by grouping geographically close nodes into clusters and by using an "hybrid" routing scheme: classically proactive approach inside each cluster and reactive approach between clusters ([7, 15, 6]). Solutions proposed to optimize broadcast lie on selecting a subset of nodes which are the only ones which forward, so reducing the number f transmitters and thus message receptions. Nevertheless, till nowadays, it seems that these two properties (routing and broadcast) have only been studied separately.

We previously proposed a clusterization algorithm for organizing a large scale multi-hop wireless network ([11]) which have revealed to be self-stabillizing ([12]), stable and robust over nodes mobility ([11]). In this report, we propose to use it also for the broadcast operation.

The remainder of this report is organized as follows. Section 2 defines the system model and introduces some notations. Section 3 presents an overview of broadcast in multi-hop wireless networks and explicit our main goals. Section 4 presents our main contribution, explaining with analyzes what lead us to use this existing structure for broadcast and how we proceed. Theoretical analysis and simulation experiments are then lead to illustrate our works. We finally compare our algorithm with other broadcast techniques. Finally, we conclude in Section 10 by discussing possible future areas of investigation.

2 System model

In a wireless multi-hop network, all nodes are alike and may be mobile. There is no base station to coordinate the activities of subsets of nodes. Therefore, all nodes have to collectively make decisions and the use of distributed algorithms is mandatory. Moreover, all communications are performed over wireless links. We classically model a multi-hop wireless network by a random geometric graph $G = (V, E)$ where V is the set of mobile nodes ($|V| = n$) and $e = (u, v) \in E$ represents a wireless link between a pair of nodes u and v if and only if they are within communication range of each other.

For the sake of simplicity, we first introduce some notations. Let's call $d(u, v)$ the distance between nodes u and v in the graph G (*i.e.* the number of hops between nodes u and v). We note $\mathcal{C}(u)$ the cluster owning the node u and $\mathcal{H}(u)$ the cluster-head of this cluster. We also note $\Gamma_k(u)$ the

k-neighborhood of a node u, *i.e.*, $\Gamma_k(u) = \{v \in V | v \neq u, d(u,v) = k\}$ and note $\delta_k(u) = |\Gamma_k(u)|$. Thus we have $\delta_1(u)$= $\delta(u)$ being the degree of node u. Note that node u does not belong to $\Gamma_k(u)$ $\forall k$.

We note $e(u/\mathcal{C}) = \max_{v \in \mathcal{C}(u)}(d(u,v))$ the *eccentricity* of a node u inside its cluster. Thus the *diameter* of a cluster is $D(\mathcal{C}(u)) = \max_{v \in \mathcal{C}(u)}(e(v/\mathcal{C}))$.

3 Related works and Goals

Two ways have to be explored. On one hand, we have solutions proposed for broadcast, but, as far as we know, they are all performed without any hierarchy. On the other hand, we have solutions to organize a multi-hop wireless network into a hierarchy for routing and managing but none seems to have been studied for broadcast.

3.1 Broadcast without hierarchy

The easiest way to broadcast a message over a network is the blind flooding, *i.e.* each node re-emits the message upon first reception of it. Obviously, this causes a great bandwidth occupation, many collisions and each node wastes its energy for receiving several copies of a single message and for transmitting it once. Therefore, this broadcast technique can not be envisaged over large scale or very dense networks.

The common goal of actual broadcast protocols in a multi-hop wireless network consists of selecting a subset of nodes which transmit the message. Indeed, the underlying purpose is to maximize the network lifetime by minimizing energy spendings. As a node spends energy while transmitting as well as receiving a packet, the main challenge is to minimize the number of these transmitters as well as the number of copies of a same message received by a node while keeping the property that every node in the network receives the packet at least once, under the assumption that the network is connected.

Many works have been realized to optimize broadcast in this purpose. Some probabilist approaches as in [13] propose that nodes forward broadcast packets with a probability *p*. The network topology is not taken into account as every node has the same probability to transmit. Some distance-based or location-based schemes ([13]) suggest that nodes forward a message only if the sender is within a distance smaller than a threshold or if the additional covered area induced by the retransmission is greater than a threshold. These methods impose that each node is able to evaluate distances and/or positions, which is not trivial in such networks. So, even if the blind flooding is outperformed, receptions rate remains high and there is no warranty that every node receives the message, even if the network is connected.

Qayyum et al. introduced the use of multi-point relay (MPR) nodes for broadcast in [16]. Each node u is aware of its 2-neighborhood and from it, selects a set of nodes among its 1-neighbors which become node u's MPR. MPR are chosen in such a way that, if u emits and only its MPR forward the message, all node $v \in \Gamma_2(u)$ receives the message. Yet, a node v forwards a message received from node u if and only if v is a MPR of node u. This gives an efficient broadcast ensuring that every node in the network receives the packet at least once when the network is connected and in an optimal number of hops if we assume an ideal MAC layer.

Other schemes have been proposed where subsets of nodes are selected for forwarding messages as some approaches based on dominating sets. Approaches based on dominating sets are inspired from the graph theory. Every node in the network is either in the dominating set (and is called an *internal node*) either 1-neighboring a node in the dominating set. As internal nodes are the ones which forward messages, every node is ensured to receive it when the network is connected. The challenge is to select these internal nodes. In [21] and [18], a simple and efficient algorithm, the NEB (*Neighbors Elimination Based*) introduces the notion of *intermediate* nodes. Node A is *intermediate* if there exist nodes B and C in $\Gamma_1(A)$ which are not direct neighbors. Two selection rules are then introduced to reduce the number of transmitter nodes. In other algorithms, dominating sets are built by electing some leaders. This election is based on local criterion as the nodes' Id ([3, 8]) or a fixed connectivity criteria (maximum degree [4]). Every node is thus either a leader, either directly linked to a leader which it joins. A group is then created, composed of a leader and every node which had joined it. Nodes which belong to several groups are called *gateways*. From it, a dominating set is composed of leader and gateway nodes. These techniques get excellent results and insure that every node is touched by the broadcast. Moreover, they are locally computed, regarding only the 2-neighborhood of nodes.

Thus, many works have been lead in order to optimize broadcast in wireless multi-hop networks but, as far as we know, none has tried to use structures already built over such networks.

3.2 Hierarchical organization

Some studies have proposed to organize networks into clusters to introduce a hierarchical routing, in order to allow scalability in wireless multi-hop networks. Indeed, over large scale, flat routing protocols (reactive or proactive) become ineffective because of bandwidth (flooding of control messages) and processing overhead (routing table computation). Introducing a hierarchy by grouping geographically close nodes into clusters and by using an "hybrid" routing scheme: classically proactive approach inside each cluster and reactive approach between clusters ([7, 15]) can solve this scalability problem. Such an organization also presents numerous advantages as to synchronize stations in a group or to attribute new service zones more easily. As far as we know, none of these structures have been studied for other purposes. All these clustering algorithms aim to identify subsets of nodes within the network and most of them bind each of these subsets to an unique leader to identify the clusters. In this case, all nodes having the same leader belong to the same cluster. Generally, nodes locally elect their cluster-head in a distributed way by using a metric to decide. This metric can be either an identity criteria (*e.g.* the lowest identity [3]), a connectivity criteria (as maximum degree [14] or density [11]) or a connectivity and identity criteria (Max-Min d-cluster [1]). When each node has to elect its parent among the nodes of its 1-neighborhood, the clusters construction leads in the same time to the formation of trees, where the roots are the cluster-heads. Nodes which have been elected by no other one become the leaves of trees. This is the case in our density-based clustering heuristic [11].

3.3 Goals

To sum up, the current efficient solutions for broadcast in multi-hop wireless networks are the ones based on MPR or dominating sets. Our main goal is thus to propose an approach based on a clusters structure introduced in [11], originally built for organizing and monitoring large scale networks. Indeed, this structure has been proved to be stable, robust and self-stabilizing ([11, 12]). Using it also for broadcast won't be more costly.

This structure is built from a cluster-head selection metric introduced in [11]: the density metric. In order to be scalable, the heuristic is completely distributed and asynchronous (avoiding any clock synchronization). The number of messages exchanges is minimized. In fact, it only uses local broadcast messages like HELLO PACKET ([5]) in order to discover the 2-neighborhood of a node.

We will see that, while building clusters, a spanning forest is constructed (Section 5.2). Thus, this process makes a selection between nodes labeling them as roots, leaves or regular nodes over the network. Thus, this heuristic creates clusters for organizing and managing a multi-hop wireless network and in the same time builds sets. We wish to use this clustering algorithm as transmitters/dominating set selection: only nodes which are non-leaves and so in the dominating set, re-transmit a broadcast packet.

4 Our contributions

In this section, we first present the metric criteria first introduced in [11] (Section 5.1) and we will see in Section 8 that this heuristic can be used at several hierarchical levels. The clusterization algorithm used in [11] in then described, stressing the clustering trees (and so the dominating sets) construction (Section 5.2). As seen in Section 3, we wish to use it as a transmitters selection: only nodes which are non-leaves re-transmit a broadcast packet.

We can then perform two kinds of broadcast: a broadcast in a cluster where the cluster-head needs to spread information over its own cluster only and a general broadcast where a single node needs to spread information over the whole network. In this second case, gateways between trees are thus needed to connect the trees and relay packets between clusters. As, by construction, every single node is connected to a non-leaf node and that the set of trees consists on a spanning forest of the network, every node is expected to receive the packet when the network is connected.

Then, we present here the trees characteristics (Section 7) to understand why the dominating sets they form can be useful for broadcast. At last, we perform and analyze the two kinds of broadcast : broadcast over a whole network (Section 9), stressing the gateways selection algorithm (Section 9.1) and broadcast in a cluster (Section 9.4) both comparing them with other broadcast algorithms.

5 The density-based heuristic

5.1 The density metric criteria

We first describe our criteria called *density* introduced in [11]. The notion of density characterizes the "relative" importance of a node in the multi-hop wireless network and within its neighborhood.

The underlying idea is that if some nodes move in $\Gamma_1(u)$ (***i.e.***, a small evolution in the topology), changes affect the microscopic view of node u (its degree $\delta_1(u)$ may change) but its macroscopic view does not evolve a lot since globally the network does not drastically change and its $\Gamma_1(u)$ globally remains the same.

The density (also noted $\rho(u)$) smooths local changes down in $\Gamma_1(u)$ by considering the ratio between the number of links and the number of nodes in $\Gamma_1(u)$.

Definition 1 (density) *The density of a node $u \in V$ is*

$$\rho(u) = \frac{|e = (v, w) \in E \mid w \in \{u\} \cup \Gamma_1(u) \; and \; v \in \Gamma_1(u)|}{\delta(u)} \tag{1}$$

To illustrate this definition, let's take the following example on Figure 1. Let's consider the node p and let's compute its *density* value $\rho(p)$. $\rho(p)$ is the ratio between the number of edges $L(p)$ and the number of nodes $|\Gamma(p)|$ in its 1-neighborhood. Nodes in the 1-neighborhood of node p are the dark gray ones ($\Gamma(p) = \{a, b, c, d, e, f\}$). We thus have $L(p)$ equal to the number of links between these nodes (dashed links) and also the number of links from node p toward these dark gray nodes (dotted links). So, $L(p) = 4 + 6 = 10$ and $\delta(p) = 6$ thus $\rho(p) = 10/6 = 5/3$. Note that to compute $\rho(p)$, node p needs to know $\Gamma_2(p)$ since it must be able to compute the number of edges that exist between all its 1-neighbors.

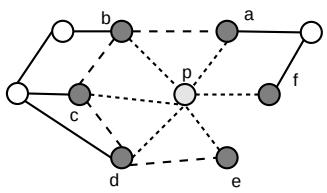


Figure 1: Density example.

5.2 Clustering tree construction

5.2.1 Basic idea

Each node locally computes its density value and regularly broadcasts it to all its 1-neighbors (***e.g.***, using `Hello` packets [5]). Each node is thus able to compare its density value to its 1-neighbors' one and decides by itself whether it joins one of them (the one with the highest density value) or it wins and elects itself as cluster-head. If there are some joint winners, the smallest Id is used to decide between them. In this way, two neighbors can not be both cluster-heads. If node u has joined node w, we say that w is node u's parent in the clustering tree – noted $\mathcal{P}(u) = w$ – and that node u is node w's child – noted $u \in \mathcal{C}h(w)$. A node's parent can also have joined another node and so on. A cluster can then extend itself until it reaches another cluster frontier. The cluster-head is the node

which has elected itself. If none of nodes has joined a node u ($\mathcal{C}h(u) = \emptyset$), node u becomes a leaf and does not belong to the dominating set. Thus, in this way, as every node chooses itself a parent among its 1-neighbors, a cluster is an oriented tree which root is the cluster-head (See illustration in Section 5.2.3). We thus build a spanning forest composed of as many trees as clusters.

To avoid that cluster-head be too off-centered in their cluster, a fusion rule is added in the case of several cluster-heads have a common neighbor. If node u is a cluster-head, all node $v \in \Gamma(u)$ must belong to $\mathcal{C}(u)$. Nevertheless, each node can join only one another node and if several cluster-heads have a common neighbor v, v has to decide between both, hence the fusion rule. The cluster-head chosen by v remains the only cluster-heads and all concerned clusters merge.

5.2.2 Heuristic

The heuristic process is quite simple. On a regular basis (frequency of HELLO packets for instance), each node computes its density value based on its view of its 2-neighborhood. This algorithm stabilizes when every node knows its *correct* cluster-head value. As proved in [12], it self-stabilizes within an expected constant and bounded time.

Algorithm 1 cluster-head selection

For all node $u \in V$
▷ *Variables initialization, only when node* u *appears.*
$\mathcal{H}(u) = \mathcal{P}(u) = u$
▷ *Checking the neighborhood*
Gather $\Gamma_2(u)$
Compute $\rho(u)$
Locally broadcast $\rho(u)$
▷ *This local broadcast can be done by piggybacking* $\rho(u)$ *in* HELLO *packets.*
▷ *At this point, node* u *is aware of all its 1-neighbors' density value and knows whether they are eligible.*
if ($\rho(u) = \max_{v \in \Gamma_1(u)}(\rho(v))$) **then**
  $\mathcal{H}(u) = u$
  ▷ u *is promoted cluster-head. If several nodes are joint winners, the winner is the one with the smallest ID.*
else
  ▷ $\exists w \in \Gamma_1(u) | \rho(w) = \max_{v \in \Gamma_1(u)}(\rho(v))$, $\rho(w) \neq \rho(u)$
  $\mathcal{P}(u) = w$
  $\mathcal{H}(u) = \mathcal{H}(w)$
  ▷ *Either* $\mathcal{P}(w) = \mathcal{H}(w) = w$ *and* u *is directly linked to its cluster-head, either* $\exists\ x \in \Gamma_1(w) \mid \mathcal{P}(w) = x$ *(w has joined another node x) and recursively* $\mathcal{H}(u) = \mathcal{H}(w) = \mathcal{H}(x)$.
end
if ($\mathcal{C}(u) = u$ and $\exists v \in \Gamma_1(u) \mid \mathcal{P}(v) \neq u$) **then**
▷ *Node* u *is a cluster-head, however all its* 1*-neighbors have not joined it.*
  if ($\mathcal{P}(v) = \mathcal{H}(v)$) **then**
  ▷ *At least two cluster-heads have a common neighbor* v*, a fusion is initiated. The winner is the cluster-head first chosen by* v *:*$nF(v)$. u *is not a cluster-head anymore.*
    $\mathcal{P}(u) = v$ and $\mathcal{C}(u) = \mathcal{C}(v)$
  else
  ▷ v *is at at least* 2 *hops away from its cluster-head, so it joins node* u
    $\mathcal{P}(v) = u$ and $\mathcal{C}(v) = u$

end
end
Locally broadcast $\mathcal{P}(u)$ and $\mathcal{H}(u)$

Note that if node u chooses node v as parent, as ID values are unique, v can not have also chosen node u as its parent. Indeed, if $\mathcal{P}(u) = v$ that means that $\rho(v) > \rho(u)$ or that $\rho(u) = \rho(v)$ and $Id(u) > Id(v)$. So, either v elects as its parent a node w such that $\rho(w) > \rho(v)$ and so $\rho(w) > \rho(u)$ and $u \neq w$, either it has to decide between itself and node u and, as it decides according to the same criterion than u, it will reach the same decision *i.e.* electing itself. Thus, this construction leads to an oriented tree as links are oriented in only one way.

As shown in [12], at the end of three message exchanges rounds, each node is aware of its parent in the tree, of its 1-neighbors' parent and whether it is a leaf, a root or a regular node in the tree. Indeed, as a node knows its 1-neighbors' parent, it is able to determine whether one of them has elected it as parent. A node is a leaf if no other node has chosen it as parent, a node is a cluster-head if it has chosen itself as parent and all its 1-neighbors have joined it; a node is a regular node if it has elected its parent among its neighbors and some nodes have joined it. In an expected constant and bounded time, every node is also aware of its cluster-head and of its neighbors' cluster-head and yet it knows whether it is a border node. Indeed, a node is a frontier node if one or several of its neighbors do not belong to the same cluster than itself.

5.2.3 Example

To illustrate this heuristic, let's run the algorithm 1 over the graph plotted on the Fig 2. In its 1-neighborhood topology, node a has two 1-neighbors ($\Gamma(a) = \{d, i\}$) and two links ($\{(a, d), (a, i)\}$); node b has 4 1-neighbors ($\Gamma(b) = \{c, d, h, i\}$) and five links ($\{(b, c), (b, d), (b, h), (b, i), (h, i)\}$). Table 1 shows the final results of density values computing.

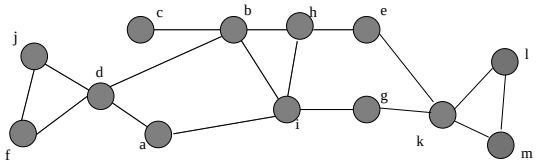
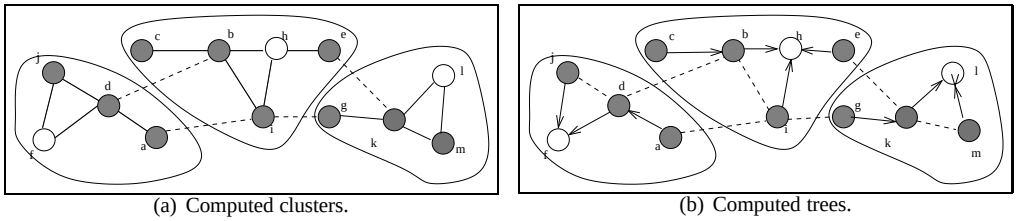


Figure 2: Example.

Nodes	a	b	c	d	e	f	g	h	i	j	k	l	m
Degree	2	4	1	4	2	2	2	3	4	2	4	2	2
# Links	2	5	1	5	2	3	2	4	5	3	5	3	3
Density	1	1.25	1	1.25	1	1.5	1	1.33	1.25	1.5	1.25	1.5	1.5

Table 1: Results of our heuristic on the illustrative example.

In our example, node c joins its neighbor node b which density is the highest ($\mathcal{P}(c) = b$). Yet, the node with the highest density in node b's neighborhood is h. Thus, $\mathcal{P}(b) = h$. As node h has the highest density in its own neighborhood, it becomes its own cluster-head: $\mathcal{H}(h) = h$ and thus: $\mathcal{H}(c) = \mathcal{H}(b) = \mathcal{H}(h) = h$. To sum up, node c joins b which joins h and all three of them belong to the cluster which cluster-head is h and so to the same tree rooted in h. Moreover, we have $\rho(j) = \rho(f)$. We thus use the ID to decide between both nodes. Let's assume that node f has the smallest Id: $\mathcal{P}(j) = f$ and $\mathcal{P}(f) = f$ so $\mathcal{H}(f) = \mathcal{H}(j) = f$. No node has chosen nodes a, j, c, e, i, g and m as parent: they become leaves. Finally, we obtain three clusters organized around three cluster-heads: node h, node l and node f (See Figure 3(a)) and in the same time, three oriented trees which roots are nodes h, l and f (See Figure 3(b)).



(a) Computed clusters. (b) Computed trees.

Figure 3: Example of clusters (b) and trees (c) computed with the density-based algorithm (cluster-heads appear in white).

Every node needs 3 messages exchanges rounds to know its parent and its condition within the tree (leaf, root or internal node). Moreover, in this example, nodes need at most 2 more exchanges rounds to know their cluster-head identity *i.e* the maximum tree depth met.

5.2.4 Density values distribution over clustering trees

Figure 4 shows the distribution of density values over clustering trees. On the left side, the density value is plotted, while the right side plots the mean difference between a node's density and its children's one, both in function of the nodes's distance to the cluster-head in the tree, for every node's children and for only non-leaf children. For instance, on Figure 4 (a), we can read that cluster-heads (nodes at distance 0) have a mean density value of 12.27. As all cluster-heads are, by construction, non-leaf nodes, both curves have the same value at this point. On Figure 4 (b), we can read that nodes at distance 0 have a density value superior of 1.48 than their children's one and 1.01 with their non-leaf children's one. These results are given here for a process intensity $\lambda = 1000$ but we could see over simulations that they do not depend on the process intensity λ.

We can thus observe, as in [10], that the strongest values are distributed around the cluster-heads and the farther a node is from one of them, the lower its density value is. Thus, the density metric creates an attraction around nodes which are elected cluster-heads. This promotes stability into the structure as claimed by the self-organizing principles ([9]). The irregularity we can note at distance 3 is due to the fusions occurring during the cluster-head election when two "winners" have a common

neighbor (see Section 5.2). We can note that except the difference between nodes at distance 0 and nodes at distance 1, the gap is enough constant. This shows that the attraction induced by the cluster-head is propagated to the end of the branches of trees. The fact that the gap is less important between cluster-heads and their children is due to the fact that all cluster-head's neighbors are actually also their children, unlike other nodes.

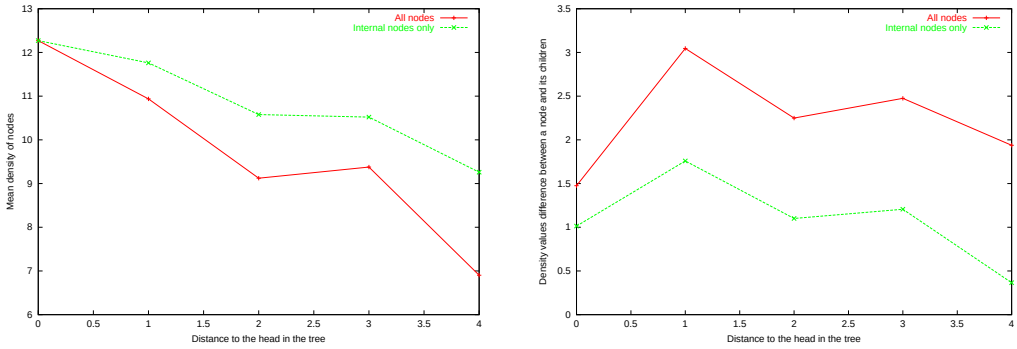


Figure 4: Density value of nodes (a) and difference between a node's density and its children's one (b) in function of the nodes's distance in tree for a process intensity of 1000.

5.2.5 Stochastic analysis of the number of cluster-heads

In [11], we analyzed the mean node's density value and clusters characteristics. Clusters appear to be robust over nodes mobility and link failures. Yet, we can expect to keep the clusters structure (and so the clustering and broadcast trees) for a pretty long time in spite of topology changes.

We also computed an upper-bound of the number of cluster-heads under Palm probability. $\mathbb{E}^o$ and $\mathbb{P}^o$ design respectively the expectation and the probability under Palm distribution. λ is the process intensity and R the transmission range. We remind that $n = |V|$.

Theorem 1 *An upper bound on the number of cluster-heads is given by:*

$$\begin{aligned}\mathbb{E}\left[\text{Number of heads in a Borel subset } C\right] &= \lambda\nu(C)\mathbb{P}^o_\Phi\left(0 \text{ is head}\right)\\ &\leq \lambda\nu(C)\left(1+\sum_{n=1}^{+\infty}\frac{1}{n}\frac{\left(\lambda\pi R^2\right)^n}{n!}\right)\exp\left\{-\lambda\pi R^2\right\}\end{aligned} \quad (2)$$

The proof of this theorem, as well as simulation results which confirm it, can be found in [10] and [11]. The basic idea is that we count under palm distribution the probability for a node to be cluster-head. This theorem is important since it shows that the number of cluster-heads does not increase with the number of nodes. Figure 5 illustrates this upper bound in function of the process intensity for different values of R. We can see that the number of clusters and so the number of trees to flood tends towards an asymptote when the number of nodes by surface unit increases. This shows

a scalability feature of this heuristic as it means that the number of gateways to compute between trees tends to an asymptote as well.

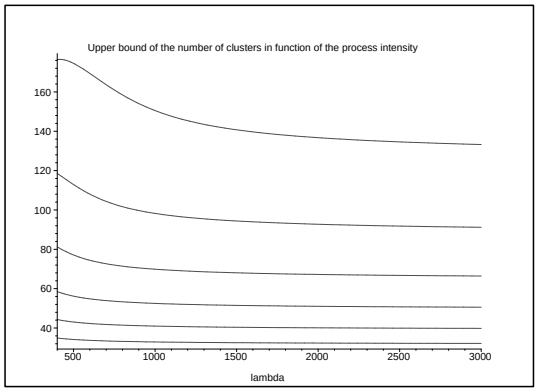


Figure 5: Upper bound for the number of clusters built by surface unit in function of the process intensity for different values of R (from the bottom to the top $R = 0.1, 0.09, 0.08, 0.07, 0.06, 0.05$ m).

6 Simulation model

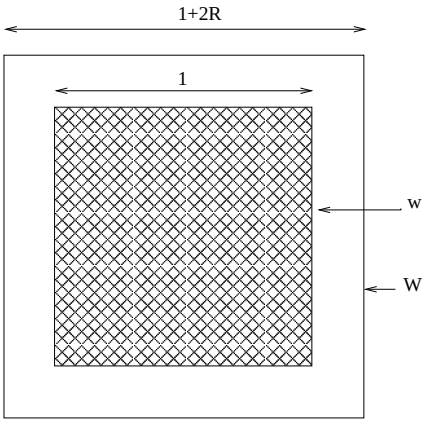


Figure 6: Only points of w are taken into account to estimate the different quantities, but the point process is generated in the square W in order to avoid the edge effects.

All simulations we performed and which are evoked in the following sections, follow the same model. We use a simulator we developed. The geometric approach used in the analysis allows to model the spatial organization of the network. Nodes are randomly deployed using a Poisson process in a $(1+2R) \times (1+2R)$ square with various levels of intensity λ (and thus various numbers of nodes). The communication range R is set to 0.1 in all tests. In each case, each statistic is the average over 1000 simulations and we fix a minimum radius and/or number of nodes such that the network is connected. When several algorithms are compared, they are compared for each iteration over the same nodes distribution.

Only the points within the square w of size 1×1 are taken into account to estimate the different quantities (mean degree, mean density, etc.). But in order to avoid border sides, the samples of the point process are generated in a larger window W. For instance, if we estimate the mean degree of the nodes ($\overline{\delta}$), we take the degree of the points in w to compute $\overline{\delta}$. If we did not consider the points of W, the points close to the edge within w would have a degree less than points close to the center introducing a bias in the estimation. Both windows are shown in Figure 6. This technique is called "minus-sampling", a more detailed description can be found in [19] page 132.

Moreover, in all our propositions and simulations, we assume an ideal MAC layer and that the algorithm is performed during a time while which the network is static.

7 Clustering trees characteristics

In this section, we describe the different characteristics of clusters and trees that we build. This allows to understand why they seem to be adapted to broadcast.

7.1 Eccentricity and depth

We first analyze mean eccentricity of nodes and cluster-heads in their cluster and the mean tree depth. The eccentricity of a node u is the greater distance in number of hops between u and any other node in $\mathcal{C}(u)$. Results expressed in number of hops are shown in Table 2.

	500 nodes	600 nodes	700 nodes	800 nodes	900 nodes	1000 nodes
# clusters/trees	11.76	11.51	11.45	11.32	11.02	10.80
$\tilde{e}(u/\mathcal{C})$	3.70	3.75	3.84	3.84	3.84	3.84
$\tilde{e}(\mathcal{H}(u)/\mathcal{C}(u))$	3.01	3.09	3.37	3.17	3.19	3.23
mean tree depth	3.27	3.34	3.33	3.34	3.43	3.51
max tree depth	5.51	5.61	5.63	5.60	5.77	5.55

Table 2: Clusters and clustering trees characteristics.

We can note that the mean tree depth is pretty low and close to the optimal we could expect which is the mean cluster-head eccentricity. The max tree depth is about constant with an increasing intensity. This also presents a good property for performing a broadcast within our clusters as this

one would thus be fast. Indeed, none node is really far away from its cluster-head and can expect to receive quickly an information spread by it. The clustering process stabilizes once each node knows its cluster ID, so their cluster-head ID, *i.e.* the root of their tree. The mean time of stabilization of the density-based algorithm is thus proportional to the depth of the trees. This one is low and so the stabilization time is.

7.2 Clusters shape

As we can see on Figure 7, the clusters built by our metric seems to match pretty well with the Voronoi diagram drawn around the cluster-heads whatever the process intensity. In a Voronoi cells graph, cells are drawn around points such that any point in a region based around point a is ensured to be closer to a in euclidean distance than any other point. This means that once cluster-heads elected, most of nodes belong to the cluster which cluster-head is the closer to them in euclidean distance among all cluster-heads.

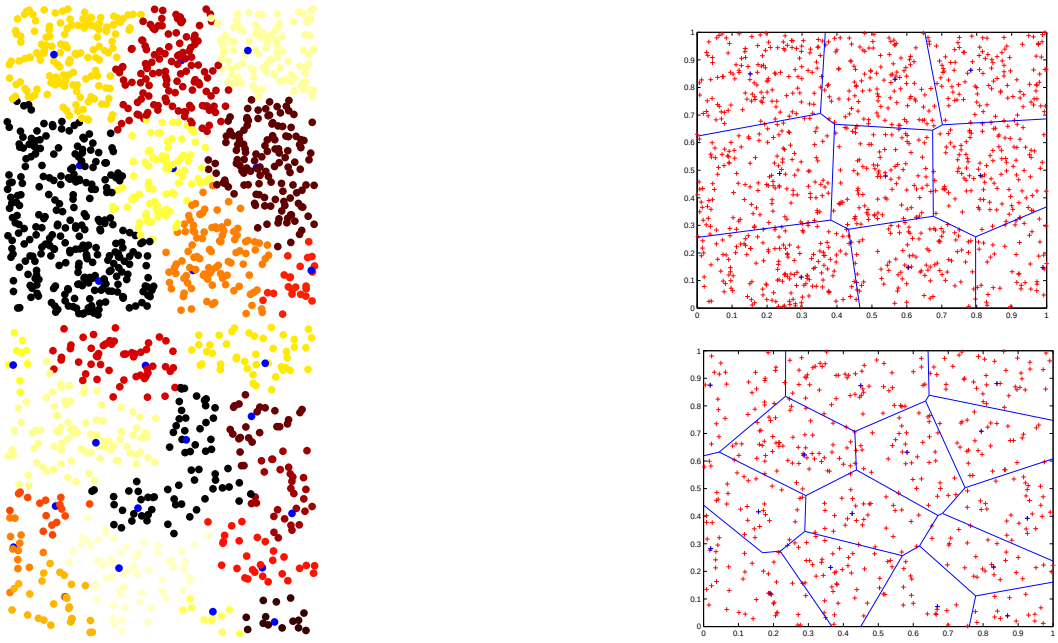


Figure 7: Density-based cluster organization (left schemes) and Voronoi diagram of cluster-heads (right schemes) for process of intensity 1000 (above) and 500 (below).

Simulations allowed us to quantify the amount of points closer to their cluster-head than any other one. As in multi-hop wireless networks, we mainly use the number of hops as distance value, we also performed similar simulations for the number of nodes, *i.e.* we computed the percentage of

nodes closer in number of hops to their cluster-head than any other one. In Figure 8, cluster-heads appear in blue, nodes laying in the Voronoi cell of their cluster-head in black, other ones in red. Results, presented in Table 3 and Figure 8, show that a great part of nodes lays in the Voronoi cell of their cluster-head whatever the process intensity. This characteristic is useful is terms of broadcast efficiency as if cluster-heads need to spread information over their own cluster, if most of nodes are closer than the one which sends the information, we save bandwidth, energy and latency.

	500nodes	600nodes	700nodes	800nodes	900nodes	1000nodes
Euclidean distance	84.17%	84.52%	84.00%	83.97%	83.82%	83.70%
Number of hops	85.43%	84.55%	84.15%	83.80%	83.75%	83.34%

Table 3: Percentage of nodes closer to their cluster-head than any other one in euclidean distance and in number of hops.

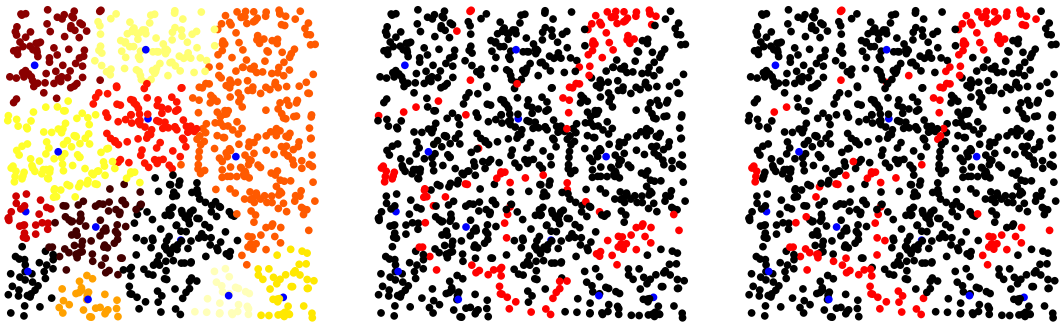


Figure 8: For a cluster organization on (a), drawing of nodes laying in the Voronoi cell of their cluster-head computed with the Euclidean distance (b) and with the number of hops (c).

7.3 Proportion of leaves

As we saw in previous sections, we intend to use our clustering trees to perform a broadcast in which only non-leaf nodes and gateways would forward a message. As the aim is to minimize the amount of transmitter nodes and useless receptions, we need a number of leaves as less as possible. Table 4 presents the proportion of leaves we have according to the process intensity.

Figure 9 shows how leaves are distributed over the graph of nodes. Leaves appear in red and internal nodes in yellow.

Yet, almost the three quarter of nodes would not emit since about 75% of nodes are leaves. Thus, compared to the blind flooding, we save about 75% of transmitters if only non-leaf nodes forward the message.

	500nodes	600nodes	700nodes	800nodes	900nodes	1000nodes
% leaves	73,48%	74,96%	76,14%	76,81%	77,71%	78,23%

Table 4: Proportion of leaves in clustering trees in function of the process intensity.

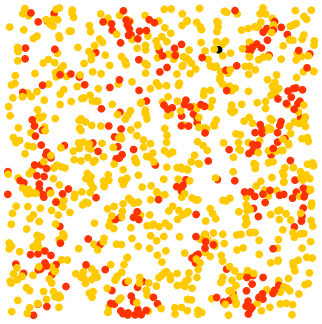


Figure 9: Distribution of leaf (red nodes) and non-leaf nodes (yellow nodes) over a clustered 1000 nodes network.

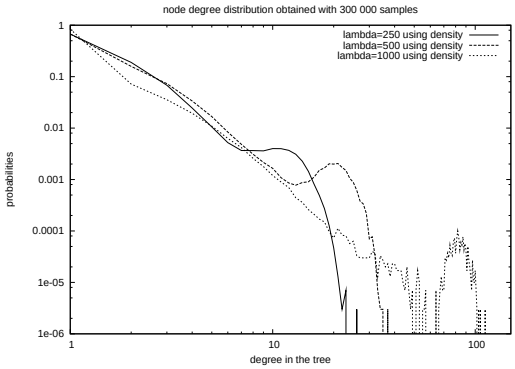


Figure 10: Distribution of node degrees in the tree for three values of λ

We also study the distribution of node degrees in the tree. In Figure 10, we plot the proportion of nodes with regard to their degrees in the tree for three different values of λ ($\lambda = 250$, 500 and 1000; $R = 0.1$). The proportion of nodes of a given degree decreasing very quickly, the curves are plotted with a log-log scale. It appears that the first part of the curves is quite linear and is thus close to a power-law distribution (we note that the rates of these curves are quite similar for the different

values of λ). This trend is broken in the second part of the curves. This is due to the degree of the cluster-heads which are notably more important than other node degrees.

The power law distribution of the node degrees shows that most of the nodes are leaves or have a very small degree and only a really few nodes have a high degree in the tree. This probabilistic behavior of node degrees have beneficial properties: when a message is broadcasted in the network, the number of transmitters will be low since it corresponds to nodes which have a degree higher than one, and the number of nodes receiving an important number of copies of the same message will be low since only a few nodes have a high degree in the tree.

7.4 Euclidean distance in clustering trees

Figure 11 plots for a node u the mean euclidean distance between u and its children in the clustering trees in function of the distance in the tree between u and $\mathcal{C}(u)$ for $\lambda = 1000$ and $R = 0.1$. Nodes at distance 0 are thus the cluster-heads, nodes at distance 1 are all nodes which have the cluster-head as parent and so on. For instance, Figure 11 shows that, in average, the cluster-head is at distance 0.064 of its children and at distance 0.055 of its non-leaf children. In a Poisson topology, the mean distance between two nodes for $R = 0.1$ and $\lambda = 1000$ is 0.066 ($0.66R$). As cluster-heads are the parents of all their neighbors, the mean distance between node at distance 0 and all their children also represents this value, what actually matches. These results are given here for $\lambda = 1000$ but we could see over simulations that they do not depend on the process intensity.

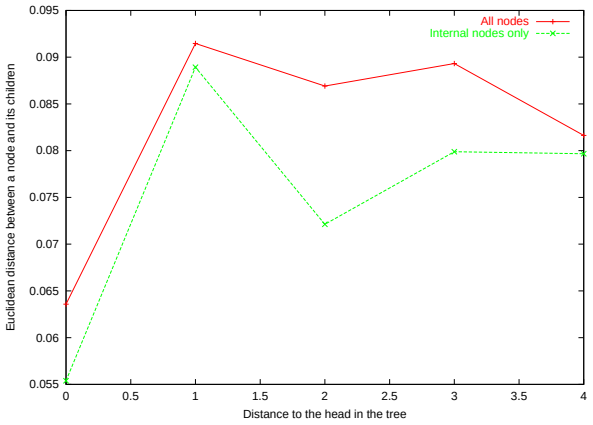


Figure 11: Euclidean distance between (1) a node and its children (2) a node and its non-leaf children in function of the nodes's distance in tree for a process intensity of 1000 and $R = 0.1$

Figure 11 shows that, in average, non-leaf nodes are closer to their parent that leaf nodes. This can be explained by the fact that nodes with the highest density in a node's neighborhood are nodes closer to the cluster-head as we saw in previous parts. The break in the curves for nodes at distance 3 is due to the fusions occurring during the cluster-head election (see Section 5.2). Moreover, we can

note that distance between a node and its non-leaf children remains higher than the average distance between two nodes ($0.6R$). This shows that nodes elect as their parent a node which is pretty far from them. This is a good property for broadcast. Indeed, let's say node u is a non-leaf node which has elected node v as parent. Both of them would forward a broadcast message as none of them are leaves and all nodes which are in the neighborhood of both of them would receive copies of the message from both u and v. The greater Euclidean distance between u and v is, the lower amount of common neighbors to u and v is and so the lower the number of useless message receptions is.

7.5 Conclusions

Yet, all these different properties and characteristics we have been able to extract from our clusters and clustering trees lead us to think they could be used in order to perform a pretty efficient broadcast over a network or/and a single cluster.

8 Hierarchical clustering

Sometimes, it could be useful to build a hierarchical structure where nodes are first gathered into clusters which are themselves gathered into upper-level clusters and so one. This can serve for applying different services according to geographic locations for instance as it is well explained in [2, 17, 20].

Let's consider a graph $G_0(V_0, E_0)$ of nodes and let's build clusters over this topology following the heuristic described in Section 5.2. We use the node's density. Let's call it level-0 density. Then, we can consider a graph $G_1(V_1, E_1)$ of an upper level, where V_1 is the set of clusters and there exists a link between $\mathcal{C}(w) \in V_1$ and $\mathcal{C}(z) \in V_1$, $w \in V$, $z \in V$ and $\mathcal{C}(w) \neq \mathcal{C}(z)$, if and only if $\exists\, u \in \mathcal{C}(w)$ and $v \in \mathcal{C}(z)$ such that $v \in \Gamma(u)$. We thus can apply our algorithm over this graph G_1 as we did over G_0. We thus have to compute the density value of nodes of G_1: level-1 density. And so on, we can apply our heuristic over a level-i graph to build clusters and form a level-$\{i+1\}$ graph. For it, we need nodes' density. Yet, we have two possibilities, either we compute a level-1 density value using our metric over this level-1 network G_1, either we use the level-0 density value of cluster-heads of G_0 as clusters are represented by their cluster-heads. Let's compare both possibilities.

Figure 12 plots the number of clusters over G_1 in function of the number of clusters over G_0 ($|V_1|$). It compares it by building a $level-2$ graph G_2 by using both kinds of density and in both cases, activating or not fusions. In our algorithm, fusions appear when two cluster-heads are distant of less than three hops. As we can see, the number of clusters over G_1 varies a lot from one kind of density to the other one. Moreover, we can note that fusions are very numerous when we compute a level-1 density value. To check this phenomena, we computed the variance between the density values. Figure 13 shows the results.

As we can note, the variance explains that fusions are so numerous when using the level-1 density. Indeed, in this case, all nodes $u \in V_1$ have approximatively the same density's value. To elect a leader, they thus need to merge and choose with another heuristic than the density value, *i.e* the node's ID. The leader election thus becomes very random and not appropriate.

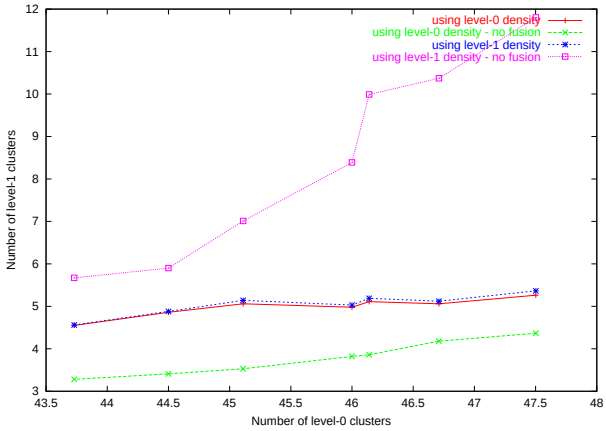


Figure 12: Number of clusters over G_1 in function of the number of clusters over G_0 ($|V_1|$).

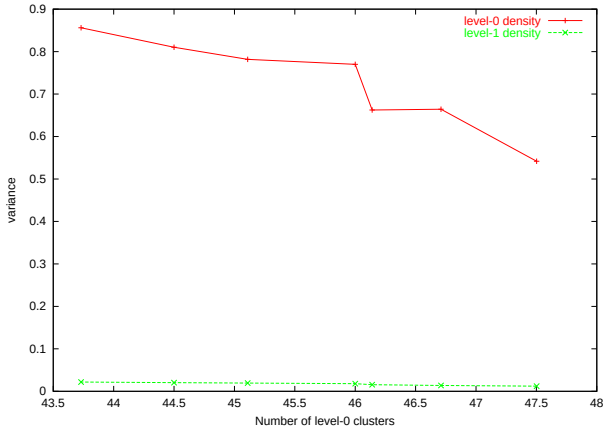


Figure 13: Variance of different kinds of density values in function of the number of clusters over G_0 ($|V_1|$).

As seen in Section 7.2, the clusters built over G_0 (V_1) are close to Voronoi cells. And, indeed, for a Voronoi cell, *(i)* the number of neighboring cells is more or less constant and equal to 6 and *(ii)* the number of links between two neighboring cells is equal to 6. This explains that the variance using the level-1 density is constant and close to 2 and that fusions are so numerous, as density value depends only on the degree of a Voronoi cell which varies very few.

Thus, our algorithm can be used for building a hierarchical topology of clusters but under the condition of using the level-0 density values to build significant clusters.

9 Broadcast over a network

In this section, we study a broadcast initiated by a random node in the network and diffused within the whole network.

9.1 Gateways selection

As we saw in previous sections, we compute a spanning forest over the network and we need to select *(i)* some gateways between neighboring clusters, *(ii)* a *mirror* node for each gateway node to relay the message in the neighboring cluster(s) and connect trees.

To illustrate this problematic, let's take the density-based clusters example plotted in Figure 3. When a message is broadcasted from node c, it is relayed within its cluster through the tree. But we need gateways to relay it towards clusters $\mathcal{C}(f)$ and $\mathcal{C}(l)$. These gateways must obviously be frontier nodes. Let's appoint node b and node i, two gateways, respectively for clusters $\mathcal{C}(f)$ and $\mathcal{C}(l)$. Thus, when nodes b and i emit, the message is received by nodes in other clusters: nodes d and g. But if it appears that these ones do not re-emit because they are leaves as g, none other nodes in cluster $\mathcal{C}(l)$ will have the message. Therefore, we also need to elect nodes, that we call *mirror-gateway*, in clusters $\mathcal{C}(f)$ and $\mathcal{C}(l)$ which belong respectively to $\Gamma(b)$ and $\Gamma(i)$ to forward the message. Let be node d for cluster $\mathcal{C}(f)$ and node g for cluster $\mathcal{C}(l)$ those nodes. Then, when nodes d and k forward the message, the packet is spread over clusters $\mathcal{C}(f)$ and $\mathcal{C}(l)$ through the clustering trees.

If $\mathcal{C}(u)$ and $\mathcal{C}(v)$ are two neighboring clusters, we note $GW(\mathcal{C}(u), \mathcal{C}(v))$ the gateway node of $\mathcal{C}(u)$ to reach $\mathcal{C}(v)$. $GWm(\mathcal{C}(u), \mathcal{C}(v))$ designs the mirror-gateway of $GW(\mathcal{C}(u), \mathcal{C}(v))$. Note that $GW(\mathcal{C}(u), \mathcal{C}(v)) \in \mathcal{C}(u)$ and $GWm(\mathcal{C}(u), \mathcal{C}(v)) \in \mathcal{C}(v)$.

9.1.1 Mirror nodes selection

As proved in [12], as the density-based clustering algorithm uses the ID as last decision criteria, every node u is aware in an expected bounded time, whether it exists among its neighbors a node v which does not belong to the same cluster than u. If so, node u is a frontier node and so is susceptible to be elected as a gateway between $\mathcal{C}(u)$ and $\mathcal{C}(v)$. Each frontier node u then selects its *mirror* among its neighbors which do not belong to $\mathcal{C}(u)$. For it, u first selects non-leaf nodes, *i.e.* a transmitter in any case and chooses among them the node with the highest density. If all node $v \in \Gamma_1(u)$ and such that $\mathcal{C}(u) \neq \mathcal{C}(v)$ is a leaf, u chooses the node with the smallest degree in order to limit the receptions. If there still exist ex-aequo, the smallest ID decides. If u is a frontier node of cluster $\mathcal{C}(v)$ ($\mathcal{C}(v) \neq \mathcal{C}(u)$), we note $m(u, \mathcal{C}(v))$ the mirror of u in $\mathcal{C}(v)$. Note that if a node u is a frontier node for several clusters different than $\mathcal{C}(u)$, it has to select a mirror for each of these clusters. For instance, in Figure 3, node i has to elect two mirrors, one in $\mathcal{C}(f)$ and one in $\mathcal{C}(l)$.

Algorithm 2 mirror selection

For all node u **such that** $\exists v \in \Gamma_1(u) \,\text{s.t.}\, \mathcal{C}(v) \neq \mathcal{C}(u)$
▷ *For each frontier node*
  For all cluster $\mathcal{C}$ **such that** $\mathcal{C} \neq \mathcal{C}(u)$ **and** $\exists v \in \Gamma_1(u) \cap \mathcal{C}$**.**
  ▷ *For each cluster for which* u *is a frontier node.*
    Select S the set of nodes such that $S = \mathcal{C} \cap \Gamma_1(u) \cap \{v \mid \mathcal{C}h(v) \neq \emptyset\}$.

▷ *u firstly selects non-leaf nodes as they are transmitters in any cases.*
if ($S \neq \emptyset$) **then**
  Select S' the set of nodes such that $S' = \{v \mid v = max_{w \in S}\rho(w)\}$.
  ▷ *u collects the non leaf nodes with the highest density in order to promote stability.*
else
  $S = \{\mathcal{C} \cap \Gamma(u)\}$.
  ▷ *All candidate nodes to be u's mirror are leaves.*
  Select S' the set of nodes such that $S' = \{v \mid v = min_{w \in S}\delta(w)\}$.
end
if ($|S'| = 1$) **then**
  ▷ *There is no conflict. S' contains only one node: node u's mirror.*
  $m(u, \mathcal{C}) = v$ such that $S' = \{v\}$.
else
  ▷ *There are conflicts. u elects the node with the smallest ID.*
  $m(u, \mathcal{C}) = v$ such that $Id_v = min_{w \in \S'} Id_w$.
end

9.1.2 Gateways nodes selection

Still according to the self-stabilization (see [12]), in an expected bounded time, each tree root r can be aware of the identity of all its neighboring clusters, the sets of the frontier nodes in $\mathcal{C}(r)$ for every neighboring cluster as well as the kind of the mirror (leaf or internal node) chosen by each of them.

The gateway selection we propose is distributed, so, a selection is performed at every step in the tree. Frontier nodes send their parent the following information: their ID, whether they are leaves and whether they have a leaf as mirror. Each parent selects the best candidate among its children and sends the same information to its own parent and so on, up to reach the cluster-head. Thus, the selection is semi-distributed as every internal node eliminate some candidates. This way, only small packets are forwarded from the frontier nodes to the cluster-head and this one does not have too many data to compute.

Figure 14 represents the mean number of children and the mean number of non-leaf children per node in function of its distance in the tree for a process intensity of 1000. These results are given here for a process intensity of 1000 but we could see over simulations that they do not depend on the process intensity. It shows that, in average, an internal node does not have a lot of non-leaf children so, at each step, the amount of data to collect is not high at all. As all cluster-heads' neighbors are also their children, cluster-heads are the only nodes with an important number of children.

Let's express that v is in the subtree rooted in u (noted $v \in s\mathcal{T}(u)$) if u is the parent of node v or if the parent of node v is in the subtree rooted in u.

$$\{v \in s\mathcal{T}(u) \cap \Gamma_1(u)\} \Leftrightarrow \{v \in \mathcal{C}h(u)\} \text{ or } \left\{v \in s\mathcal{T}(u) \cap \bar{\Gamma}_1(u)\right\} \Leftrightarrow \{\mathcal{P}(v) \in s\mathcal{T}(u)\}$$

The best candidate choice is performed as follows. For each of the neighboring clusters of its subtree, an internal node u considers the set S of the candidate nodes (frontier nodes) ($S = \{v \in s\mathcal{T}(u) \mid \exists w \in \Gamma(v) \mid \mathcal{C}(w) \neq \mathcal{C}(u)\}$). Then, it selects among them the subset $S' \subset S$ of internal nodes, still in order to limit the number of transmitter nodes ($S' = \{v \in S,\ \mathcal{C}h(v) \neq \emptyset\}$). If $S' = \emptyset$, the selection is processed among nodes in S ($S' = S$) as S is thus only composed of leaves.

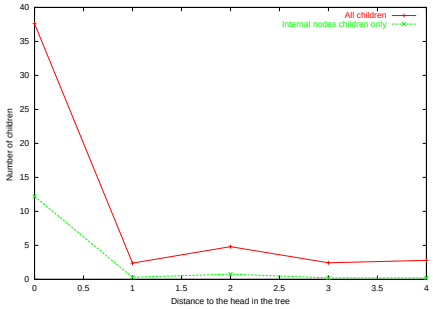


Figure 14: Number of children of nodes in function of their distance in the clustering tree.

From there, u selects the subset $S'' \subset S'$ of nodes which mirror is a non-leaf node.
$S'' = \{v \in S' \mid S(m(v, \mathcal{C}(w))) \neq \emptyset\}$.
If $S'' = \emptyset$, the selection is processed among nodes in S' ($S'' = S'$).

At last, it chooses as better candidate the node $v \in S''$:

- with the highest density if S'' is composed of non-leaf nodes, in order to promote stability:
  $GW(\mathcal{C}(u), \mathcal{C}(v)) = \{v \in S'' \mid \rho(v) = max_{w \in S''} \rho(w)\}$
- with the lowest degree if S'' is only composed of leaves, as when a node emits, all its neighbors hear it, if we minimize the number of neighbors, we limit the number of receptions.
  $GW(\mathcal{C}(u), \mathcal{C}(v)) = \{v \in S'' \mid \delta(v) = min_{w \in S''} \delta(w)\}$

At the end, if there still exist some joint winners, the one with the lowest ID is elected.

To illustrate this selection, let's take the example of Figure 3. Node f's cluster has only one neighbor $\mathcal{C}(h)$ and two candidate nodes for being gateway: a and d. As these nodes know they are frontier nodes, they choose their mirror. a and d have both only one possibility for their *mirror*. Thus we have $m(a, \mathcal{C}(h)) = i$ and $m(d, \mathcal{C}(h)) = b$. This information is relayed by a to its parent: node d. Node d decides which better suits as gateway between itself and its child. As a is a leaf and d an internal node, d elects itself as gateway. It sends this information to its parent which also is the cluster-head: f. This one does not have any selection to do as d is the only candidate it receives. We thus have $GW(\mathcal{C}(f), \mathcal{C}(h)) = d$.

In the same time, h selects its own gateways $GW(\mathcal{C}(h), \mathcal{C}(f))$ between $\mathcal{C}(h)$ and $\mathcal{C}(f)$ and $GW(\mathcal{C}(h), \mathcal{C}(l))$ between $\mathcal{C}(h)$ and $\mathcal{C}(l)$. Candidates for $GW(\mathcal{C}(h), \mathcal{C}(f))$ are nodes b and i. As both of them have only one possibility for their mirror, we have $m(b, \mathcal{C}(f)) = d$ and $m(i, \mathcal{C}(f)) = a$. b is an internal nodes, i a leaf. So $\mathcal{H}(h)$ decides $GW(\mathcal{C}(h), \mathcal{C}(f)) = b$. Candidates for $GW(\mathcal{C}(h), \mathcal{C}(l))$ are nodes e and i, $m(e, \mathcal{C}(l)) = k$ and $m(i, \mathcal{C}(l)) = g$. Nodes e and i are both leaves but $m(e, \mathcal{C}(l))$ is an internal node unlike $m(i, \mathcal{C}(l))$. $\mathcal{H}(h)$ thus elects node e as gateway for cluster $\mathcal{C}(l)$: $GW(\mathcal{C}(h), \mathcal{C}(l)) = e$. We can note that, in this particular case, cluster $\mathcal{C}(h)$ could have elected only one gateway for both neighboring clusters: node i. Nevertheless, in general cases, to identify such situation, we need a centralized view. That means that cluster-heads would need to have all information about all its frontier nodes. This would use a lot more of bandwidth and calculus resources

on few nodes (the cluster-heads). Furthermore, this would be useless in most cases. Moreover, electing several gateways does not add cost in every cases as, as in our example, some internal nodes are chosen, and adds redundancy and so reliability.

Note that gateways are chosen by cluster-heads while the mirror-gateways are chosen and maintained by gateway nodes only. Also note that, if $\mathcal{C}(u)$ and $\mathcal{C}(v)$ are two neighboring clusters, $GW(\mathcal{C}(u), \mathcal{C}(v)) \neq GW(\mathcal{C}(v), \mathcal{C}(u))$ in most cases.

9.1.3 Gateways analysis

If we consider two neighboring clusters $\mathcal{C}(u)$ and $\mathcal{C}$(v), we may have four types of gateway between them:

- Leaf - Leaf gateways: $GW(\mathcal{C}(u), \mathcal{C}(v))$ and $GWm(\mathcal{C}(u), \mathcal{C}(v))$ are both leaves. This kind of gateway is the more costly as it adds two transmitter nodes and thus induces more receptions.
- Leaf - Internal Node gateways: $GW(\mathcal{C}(u), \mathcal{C}(v))$ is a leaf and $GWm(\mathcal{C}(u), \mathcal{C}(v))$ an internal node. This kind of gateway adds only one transmitter node. It's the less current, as shown later by simulations.
- Internal Node - Leaf gateways: $GW(\mathcal{C}(u), \mathcal{C}(v))$ is an internal node and $GWm(\mathcal{C}(u), \mathcal{C}(v))$ a leaf. This kind of gateway adds only one transmitter node.
- Internal Node - Internal Node gateways: $GW(\mathcal{C}(u), \mathcal{C}(v))$ and $GWm(\mathcal{C}(u), \mathcal{C}(v))$ are both internal nodes. This kind of gateway is the one we try to favor since it does not add extra-cost as it does not add any transmitter neither induces any additional receptions. But, as we will see, they unfortunately are the less current ones.

	500 nodes	600 nodes	700 nodes	800 nodes	900 nodes	1000 nodes
#clusters	11.93	11.64	11.36	11.30	11.14	10.72
# gw per cluster	5.86	6.02	6.16	6.20	6.22	6.26

Table 5: Number of gateways to select per cluster in function of the intensity process.

Some characteristics of the gateways we have to select to connect our trees are given by Table 5 and Figure 15 .

Table 5 shows the mean number of gateways a cluster has to elect. As we can note, this number is not very high and remains almost constant while the process intensity increases. This shows a scalability feature of this heuristic.

Figure 15 gives the proportion of each kind of gateways. We can note that the two less met kinds of gateways are the Leaf - Internal Node and Internal Node - Internal Node gateways. This is due to the fact that, by construction, most of frontier nodes are leaves. This also explains the great proportion of other kinds as, as soon as there exists an internal node as frontier nodes, it is elected. The more sparse the network is, the less chance we have to find internal nodes on borders. So, the

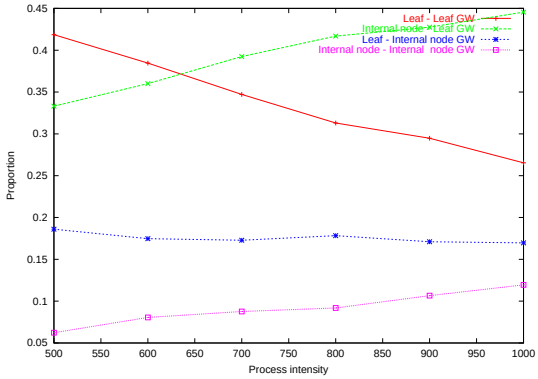


Figure 15: Proportion of each kind of gateways in function of the process intensity.

proportion of Internal node - Leaf gateways increases with the intensity process while the proportion of Leaf-Leaf gateways decreases.

Note that when a broadcast is performed, all gateways are not necessary used as between two neighboring clusters $\mathcal{C}(u)$ and $\mathcal{C}(v)$, there exist two gateways $GW(\mathcal{C}(u),\mathcal{C}(v))$ and $GW(\mathcal{C}(v),\mathcal{C}(u))$ and in most cases we use only one of them.

9.2 Broadcast heuristic

When a broadcast is performed over the whole network, node u forwards a packet received from node v if

(i) it is the first time it receives it AND it is not a leaf.

(ii) it is the first time it receives it AND $\mathcal{C}(u) = \mathcal{C}(v)$ AND $u = GW(\mathcal{C}(u),\mathcal{C}(w))\forall w \in V$.

(iii) it is the first time it receives it AND $\mathcal{C}(u) \neq \mathcal{C}(v)$ AND $u = GWm(\mathcal{C}(v),\mathcal{C}(u))$.

Note that a mirror-gateway $GWm(\mathcal{C}(v),\mathcal{C}(u))$ node forwards a message coming from $\mathcal{C}(v)$ whatever the transmitter node in $\mathcal{C}(v)$ is and which is not necessarily the gateway $GW(\mathcal{C}(u),\mathcal{C}(w))$. This adds robustness in case of the link between $GW(\mathcal{C}(u),\mathcal{C}(w))$ and $GWm(\mathcal{C}(u),\mathcal{C}(w))$ fails.

9.3 Broadcast simulations and analysis

9.3.1 Analysis

In this section, we first analyze the number of messages received by a typical node for a given broadcast. We give two formulae of the mean number of receptions. These formulae are expressed for a generic random graph under minimal hypothesis and for the geometric model which uses a Poisson point process to describe the location of the nodes. For the last case, we use Palm calculus to derive the mean number of receptions perceived by a typical node.

Random graph We consider a random graph $G(V, E)$. Let be $Relay$ (the set of relays) a subset of the vertice of G. We define

- N as the random variable which represents the number of vertice in G ($N = |V|$),
- for $u \in G$, $\delta_R(u)$ as the number of relays in the neighborhood of u,
- $\overline{r}$ as the mean number of receptions per node,
- and Z as the total number of receptions generated by a broadcast.

We remind that, for $u \in V$, $\delta(u)$ is the number of neighbors of u (in other words the degree).

We do the following hypothesis:

- the subset $Relay$ forms a connected dominating set of G,
- the degrees of the nodes are identically distributed,
- the degrees of the relays in G are identically distributed,
- the number of receptions of the nodes is identically distributed.

Only the relays broadcast the message, therefore, the number of receptions of a node (relay or not) corresponds to the number of relays in its neighborhood. We have,

$$\overline{r} = \mathbb{E}[\delta_R(u)], \forall u \in V$$

Since the number of receptions is equi-distributed, we have

$$\overline{r} = \mathbb{E}\left[\frac{Z}{N}\right]$$

Z can be written in two ways:

$$Z = \sum_{u \in V} \delta_R(u)$$

and

$$Z = \sum_{v \in R} \delta(v) = \sum_{v \in G} \delta(v) \mathbb{1}_{v \in R}$$

We take this last equality to compute $\mathbb{E}\left[\frac{Z}{N}\right]$. We condition this quantity by the different values of N.

$$
\begin{aligned}
\overline{r} &= \mathbb{E}\left[\frac{Z}{N}\right] \\
&= \mathbb{E}\left[\frac{\sum_{v\in V}\delta(v)\mathbb{1}_{v\in R}}{N}\right] \\
&= \sum_{k=1}^{+\infty}\mathbb{E}\left[\frac{\sum_{i=1}^{k}\delta(v_i)\mathbb{1}_{v_i\in R}}{k}\Big|N=k\right]\mathbb{P}(N=k) \\
&= \sum_{k=1}^{+\infty}\sum_{i=1}^{k}\frac{1}{k}\mathbb{E}\left[\delta(v_i)\mathbb{1}_{v_i\in R}\right]\mathbb{P}(N=k) \\
&= \sum_{k=1}^{+\infty}\mathbb{E}\left[\delta(v_1)\mathbb{1}_{v_1\in R}\Big|N=k\right]\mathbb{P}(N=k) \\
&= \mathbb{E}\left[\delta(v_1)\mathbb{1}_{v_1\in R}\right] \\
&= \mathbb{E}\left[\delta(v_1)\Big|v_1\in R\right]\mathbb{P}(v_1\in R)
\end{aligned}
$$

In the last equalities, v_1 is a node arbitrary chosen among the set of vertice in G (a typical node).

Poisson Point Process In this paragraph, we present the same formulae but deduced from Palm calculus. Let Φ be a homogeneous Poisson Point Process of intensity λ ($\lambda > 0$) distributed in the plane. Let Φ_{Relay} be a thinning of Φ. The points of Φ_{Relay} represent the relays. We note that Φ_{Relay} is not a priori an independent thinning of Φ and thus, is not a priori a Poisson point process. However, we suppose that Φ_{Relay} is still a stationary point process of intensity λ_{Relay}. We assume that two points (x, y) of Φ are connected if and only if the Euclidean distance between x and y is lower than R ($d(x, y) \leq R$). We denote by $\Phi(S)$ (resp. $\Phi_{Relay}(S)$) the number of points of the points process Φ (resp. Φ_{Relay}) laying in the surface S. $B(x, R)$ stands for the ball of radius R centered in node x.

We consider the only nodes/points within an observation window W, which is a square of size $L \times L$ with $L \in I\!R^+$. Since the Poisson point process is distributed in $I\!R^2$, nodes of W may receive the broadcast from node outside W.

For a typical point, *i.e.* the point in 0 under Palm probabilities, the mean number of receptions corresponds to the mean number of points of Φ_{Relay} at distance less than R. If $\overline{r}$ is the mean number of receptions per node,

$$\overline{r} = \mathbb{E}^o_\Phi\left[\Phi_{relay}(B_0^{'})\right]$$

where $\mathbb{E}^o_\Phi$ is the expectation under palm probabilities *w.r.t.* the process Φ and $B_x^{'} = B(x, R)\backslash\{x\}$. According to the Mecke Formula (see [19]), the total number of receptions Z received by the nodes within W is

$$\mathbb{E}\left[\int_W \Phi_{Relay}(B_x^{'})\Phi(dx)\right] = \lambda \mathbb{E}_{\Phi}^{o}\left[\Phi_{Relay}(B_0^{'})\right]$$

By stationarity of the two point processes Φ and Φ_{Relay}, we have,

$$\mathbb{E}\left[\int_W \Phi_{Relay}(B_x^{'})\Phi(dx)\right] = \mathbb{E}\left[\int_W \Phi(B_x^{'})\Phi_{Relay}(dx)\right]$$

The left hand side of the equality is the total number of receptions perceived by the nodes within W (the relay can be outside) and the right hand side is the total number of receptions perceived by the whole points of Φ but generated by the relays standing in W. Applying the Mecke formula to both sides of the equality, we have:

$$\lambda \mathbb{E}_{\Phi}^{o}\left[\Phi_{Relay}(B_0^{'})\right] = \lambda_R \mathbb{E}_{\Phi_{Relay}}^{o}\left[\Phi(B_0^{'})\right]$$

and,

$$\overline{r} = \mathbb{E}_{\Phi}^{o}\left[\Phi_{Relay}(B_0^{'})\right] = \mathbb{E}_{\Phi_{Relay}}^{o}\left[\Phi(B_0^{'})\right]\mathbb{P}_{\Phi}^{o}\left(0 \in \Phi_{Relay}\right) = \frac{\lambda_{Relay}}{\lambda}\mathbb{E}_{\Phi_{Relay}}^{o}\left[\Phi(B_0^{'})\right]$$

This last formula corresponds exactly to the results obtained in the previous paragraph. It may be interpreted as follows: the mean number of receptions per node is the product of the degree of a relay and of the probability for a node to be a relay (or equivalently the mean ratio of relays/nodes). An efficient relays selection must choose relays in a way to minimize this product. In the next sections, we shall use this result to compare different relays selection algorithms and their impact on the mean number of receptions. We shall show that they minimize either the degree of the relays (for instance for the MPR) or the number of relays used (for instance with our clustering algorithm).

Remark 1 *If we just consider points of W as relays and receivers, the total number of receptions becomes:*

$$\begin{aligned}
\mathbb{E}\left[\int_W \Phi_{Relay}(B_x^{'}\cap W)\Phi(dx)\right] &= \mathbb{E}\left[\int_W \Phi(B_x^{'}\cap W)\Phi_{Relay}(dx)\right] \\
&= \lambda_{Relay}\int_W \mathbb{E}_{\Phi_{Relay}}^{o}\left[\Phi(B_0^{'}\cap(W-x))\right]dx \\
&= \lambda\int_W \mathbb{E}_{\Phi}^{o}\left[\Phi_{Relay}(B_0^{'}\cap(W-x))\right]dx
\end{aligned}$$

Unfortunately, neither the degree of relays nor the proportion of relays can be found for the considered algorithms but the blind flooding for which the results are trivial. However, these quantities can be evaluated by simulations. It is the goal of the next sections.

9.3.2 Number of receptions and transmitters

In order to evaluate our algorithm, we compare it to other broadcast techniques as blind flooding, Neighbors Elimination-Based [18] and Multi-Point Relay [16].

The broadcast is initiated by a randomly-chosen source over the whole network. Significant characteristics we first note are the proportion of nodes which need to re-emit the message and the mean number of copies of the broadcasted message that a node receives. As the main goal is to limit energy spending and bandwidth occupation in order to maximize the network lifetime, these values have to be as small as possible, keeping the property that every node receives the packet at least once when the network is connected. As shown above, this quantity depends on the relays degree and the proportion of transmitters. These two quantities are also used to compare the different kinds of relays selection.

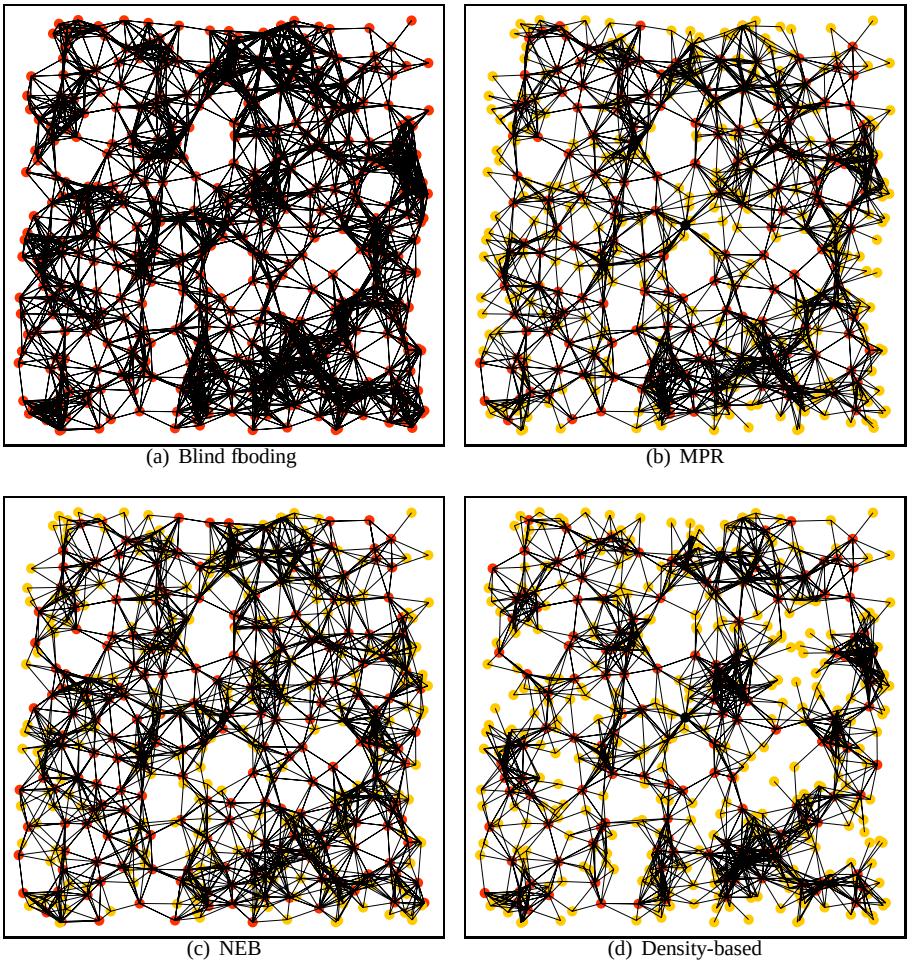
Figure 9.3.2 plots an instance of broadcast trees over a 500 nodes network computed with the different algorithm. The source appears in black, transmitters in red and leaves in yellow. The black lines show the paths followed by the message.

As Figure 17 plots, when a broadcast is initiated in the network by a random node, our algorithm induces less re-transmissions and less receptions than other metrics. Thus, it spends less energy and resources.

In Figure 18(a), we draw the mean degree of the relays when the different broadcast techniques are used. We observe that the density-based relays selection maximizes the mean degree of the relays unlike the NEB technique for which the mean degree of transmitters is smaller than the mean degree of nodes expressed by the blind flooding. Since the mean node density value is almost proportional to the mean node degree (see [10]), density-based selection elects nodes with a high degree. We also note that MPR selection elects transmitters without favoring nodes with small or high degree.

In Figure 18(b), we show the proportion of transmitters used in a broadcast. For a node, this ratio corresponds to the probability to be a relay. Note that, in bind flooding, as every node forwards the message once, the proportion of transmitters is always 1. All algorithms use notably less relays than blind flooding and this proportion decreases with the mean degree of nodes (intensity of the process). There is an economy of scale when the process intensity increases: when new points appear, there is no need of new transmitters since the square is already covered by transmitters. If we compare the proportion of relays for the different metrics, it appears that it is the density metric which notably minimizes the number of relays. With this metric, we then have a small number of transmitters with high degree. But, Figure 17 shows that it is this algorithm which offers the best performance with regard to the mean number of receptions per node. It would be interesting to compare these results with optimal selections of relays found in a centralized way. That could allow us to confirm that algorithms selecting a small number of transmitters with high degree give best results in terms of performance. We reserve this task for future works.

We can note that, as while a broadcast is performed, nodes have a label which indicates whether they have to forward the packet. This decision does not depend on the paths the broadcast uses or other parameter. Thus, these results does not depend on the position of the source in the clustering tree.



(a) Blind fboding (b) MPR

(c) NEB (d) Density-based

Figure 16: Broadcast trees computed with different metric over a $\lambda = 500$ network for a centered source.

9.3.3 Latency

Thus, we showed that the density-based relays selection minimizes the mean number of receptions. However, other performance metrics may be taken into account. For instance, we wondered about the induced latency, *i.e.* how much time we need to insure that every node has received the packet. Since in the MPR selection, relays are selected in order to reach the 2-neighborhood after two hops, the k-neighborhood of the node beginning the broadcast is reached within k hops. As we consider

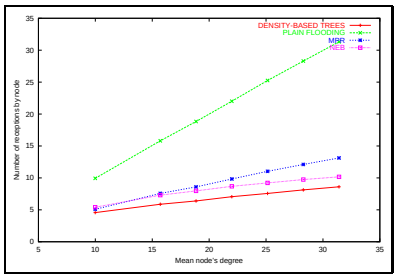
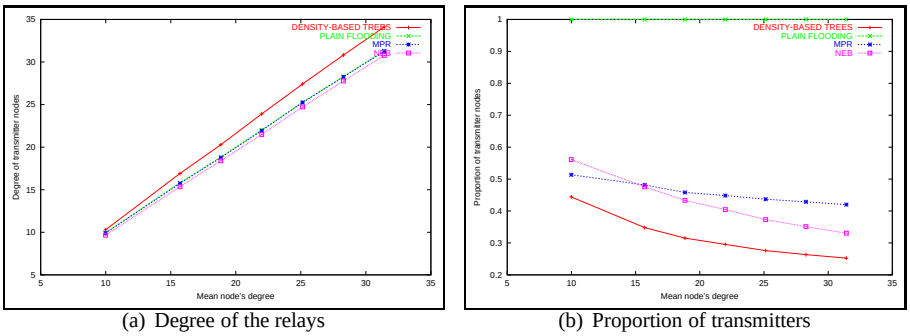


Figure 17: Number of receptions by node in fct of the nodes degree.



(a) Degree of the relays (b) Proportion of transmitters

Figure 18: Mean degree of relays and Proportion of transmitters in function of the mean nodes degree ($\lambda\pi R^2 - 1$) *w.r.t.* the different metrics.

an ideal MAC layer, MPR gives the optimal results. We thus compare our heuristic to the MPR one to measure how far we are from the optimal solution. We consider a time unit as a transmission step (*i.e.* 1 hop). Table 6 presents results. "MAX" values represent the time needed for every node to receive the packet at least once. "MEAN" values represent the mean time a node has to wait till the first reception of the packet.

Yet, we can note that, even if our algorithm is not optimal, results are not so far. Figure 19 represents the propagation in time for a broadcast initiated by a centered source at time 0. Clusterheads appear in blue and source in green. The color of other nodes depends on the time they receive the broadcast. The darker the color is, the shorter the time is.

On Figure 19 (a), we can observe with concentric circles that MPR effectively performs a broadcast within an optimal time. The difference between radius of two following circles corresponds to the nodes transmission range: R.

	500 nodes		600 nodes		700 nodes	
	MEAN	MAX	MEAN	MAX	MEAN	MAX
MPR	5.13	8.97	4.98	8.83	4.88	8.40
Density	6.31	11.05	6.21	10.82	6.22	10.78
	800 nodes		900 nodes		1000 nodes	
	MEAN	MAX	MEAN	MAX	MEAN	MAX
MPR	4.88	8.40	4.81	8.23	4.78	8.07
Density	6.24	10.95	6.15	10.66	6.19	10.74

Table 6: Mean and Max time for receiving the message.

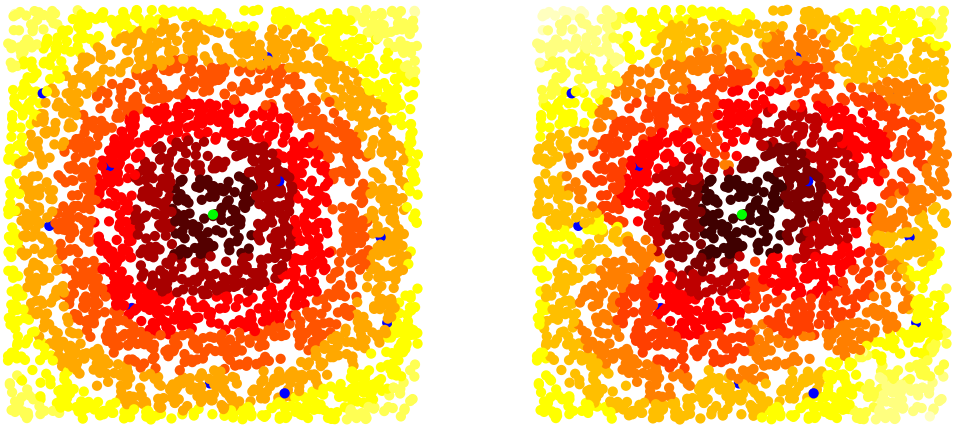


Figure 19: Propagation time of a packet broadcasted by a centered source (a) using MPR (b) using density-based clustering trees.

9.3.4 Gateways utilization

In this section, we analyze the proportion of gateways used. Indeed, among all gateways selected in Section 9.1, all are not needed when a single broadcast is performed. Table 7 shows the number of gateways used per cluster when a broadcast is initiated by a randomly chosen source. We can note that this number is quite constant when the intensity of nodes increases and that the amount of used gateways remains pretty low, always comprised between 1 and 2. This means that generally, in most cases, either the broadcast enters a cluster and dies in it (in this case, it uses only one gateway), either it traverses it and thus uses two gateways (one to enter the cluster, one to leave it).

Figure 20 shows the proportion of each kind of gateways which are used. As we can see, most of used gateways are the ones which add only one transmitter node. This is true even for small

intensities when the number of "Leaf-Leaf" gateways is the highest. This shows that our algorithm can adapt and favor internal nodes naturally.

	500nodes	600nodes	700nodes	800nodes	900nodes	1000nodes
#clusters	11.93	11.64	11.36	11.30	11.14	10.72
#gw used per cluster	1.76	1.74	1.73	1.76	1.68	1.66

Table 7: Number of gateways used per cluster when a randomly chosen source broadcasts a packet over the network in function of the intensity process.

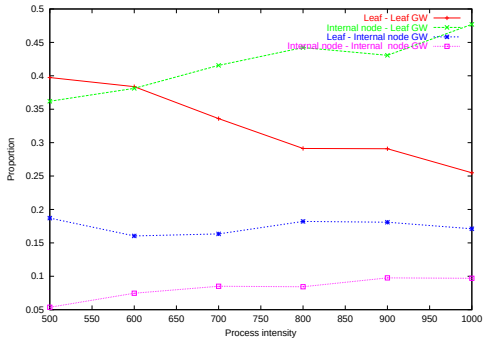


Figure 20: Proportion of each kind of gateways used in function of the process intensity.

9.4 Broadcast in a cluster

In this section, we study a broadcast initiated by a node in a cluster $\mathcal{C}(u)$ and diffused within $\mathcal{C}(u)$ only.

9.4.1 Broadcast heuristic

When broadcasts in clusters are performed, node u forwards a packet received from node v if, and only if the three following conditions hold:

- it is the first time it receives it
- $\mathcal{C}(v) = \mathcal{C}(u)$ (The message comes from its cluster.)
- $\mathcal{C}h(u) \neq \emptyset$ (u is an internal node)

In this case, there is no notion of gateway as clusters are trees and thus connected.

9.4.2 Number of receptions and transmitters

As we did in Section 9, we compare our broadcast algorithm to blind flooding, NEB and Multi-Point Relays [16], but this time we limit it within a cluster. We suppose that a broadcast is performed in each cluster, initiated by cluster-heads. Significant characteristics we noted are the proportion of nodes which need to re-emit the message and the mean number of copies of the broadcasted message that a node receives.

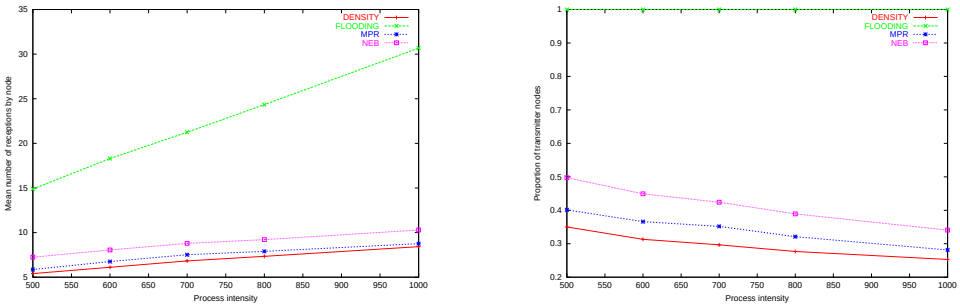


Figure 21: Mean number of receptions by node (a) and Proportion of transmitter nodes (b) for a broadcast in a cluster in function of the process intensity and the metric used.

Figure 21 shows the results. We can see that our broadcast algorithm still obtains the best results in terms of receptions and transmitters ratio. This kind of broadcast might be interesting for clustered architectures when, for instance, a cluster-head needs to spread information in its cluster like in sensors networks for instance where the base station may need to update devices or spread a query over them. Other broadcast algorithms are not designed for that kind of broadcast and so, it was predictable that they do not obtain excellent results. Nevertheless, as no broadcast scheme exists for that kind of broadcast to our knowledge, they can give us a reference.

Remark that the broadcast has been initiated by cluster-heads but as, in a cluster/tree, all internal nodes forward, results regarding the mean number of receptions per node and transmitters ratio induced by our algorithm would be the same if the broadcast was initiated by any node in the cluster.

9.4.3 Latency

Then, like in Section 9.3.3, we wondered about the induced latency, *i.e.* how much time we need to insure that every node has received the packet. For it, we still compare to the MPR algorithm as, even if MPR is not originally to be used in such broadcast, it still gives the optimal solution. We thus compare our heuristic to the MPR one to measure how far we are from the optimal solution. We still consider a time unit as a transmission step, *i.e.* 1 hop. Table 8 presents results. "MAX" values still represent the time needed for every node to receive the packet at least once and "MEAN" values the mean time a node has to wait till the first reception of the packet.

Yet, we can note that, even if our algorithm is not optimal, results are very close. For mean values, in average, our algorithm needs 0.5 steps more than the optimal one. This also shows that,

	500 nodes		600 nodes		700 nodes	
	MEAN	MAX	MEAN	MAX	MEAN	MAX
MPR	1.76	4.71	1.78	4.75	1.78	4.85
Density	1.80	5.08	1.83	5.23	1.83	5.38
	800 nodes		900 nodes		1000 nodes	
	MEAN	MAX	MEAN	MAX	MEAN	MAX
MPR	1.81	4.83	1.81	4.80	1.82	5.00
Density	1.87	5.29	1.87	5.50	1.88	5.30

Table 8: Mean and Max time for receiving the message broadcasted in a cluster by the cluster-head.

even if the routes in trees from the cluster-heads to other nodes in their cluster are not always the shortest ones, they are very close to them.

9.4.4 Trees utilization

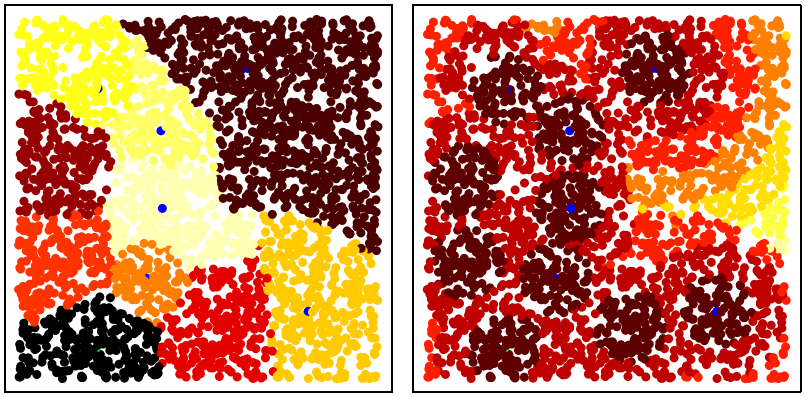
Table 9 gives the proportion of nodes which receive the packet for the first time by their parent or by one of their children. This feature shows wether the message which is broadcasted by the cluster-head follows the branches of the trees. Indeed, a node u always receives the message by its parent (as all non-leaf nodes forward) but it could have received it before from another way as paths are not always optimal. In this case, the shorter route between u and its cluster-head is not by following the route traced by the tree. We can thus see that routes are the shortest ones in number of hops for more than 70.00% of the cases. We can notice that, as the message is going down the tree, none of nodes receives it for the first time by one of their children.

	500nodes	600nodes	700nodes	800nodes	900nodes	1000nodes
% by parent	78.74%	76.81%	74.57%	73.21%	71.31%	70.13%
% by a child	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%

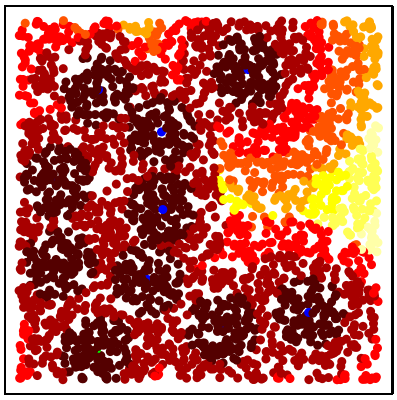
Table 9: Proportion of nodes which receive the first packet by their parent or by one of their children.

10 Conclusion and perspectives

We have studied our previous clustering architecture by the use of theoretical and simulation tools. This allowed us to outline some features which lead us to propose a new scheme for selecting transmitter nodes used to reduce the cost of broadcast messages in multi-hop wireless networks. The resulting broadcast has been shown to outperform existing broadcast solutions regarding the number of transmitters and messages receptions induced. It is then analyzed and compared to existing techniques.



(a) Clusters (b) Propagation with the MPR



(c) Propagation with the density metric

Figure 22: Propagation time of a packet broadcasted by cluster-heads of clusters plotted in *a* using MPR *(b)* or density-based trees *(c)*.

In future, we intend to test deeper this broadcast algorithm in term of complexity, stability over nodes mobility and links failures. We also intend to improve the broadcast trees by introducing some rules transforming internal nodes into leaves in order to reduce the number of transmitters and thus to save more energy.

References

[1] A. Amis, R. Prakash, T. Vuong, and D. Huynh. Max-min d-cluster formation in wireless ad hoc networks. In *Proceedings of the IEEE INFOCOM*, march 2000.

[2] F. Baccelli, M. Klein, M. Lebourges, and S. Zuyev. Géométrie aléatoire et architecture de réseaux de communications. Research Report RR-2542, INRIA, 1995.

[3] P. Basu, N. Khan, and T. Little. A mobility based metric for clustering in mobile ad hoc networks. In *Proceedings of Distributed Computing Systems Workshop 2001*, 2001.

[4] G. Chen and I. Stojmenovic. Clustering and routing in mobile wireless networks. Technical Report TR-99-05, SITE, June 1999.

[5] T. Clausen and P. Jacquet. Optimized link state routing protocol (olsr). RFC 3626, October 2003.

[6] Z. Haas, M. Pearlman, and P. Samar. Zone routing protocol (zrp). Internet Draft, July 2002.

[7] M. Hayashi and C. Bonnet. A new method for scalable and reliable multicast system for mobile networks. In *1st Networking - ICN 2001 : First International Conference*, Colmar, France, July, 9-13th 2001.

[8] M. Jiang and Y. Tay. Cluster based routing protocol(CBRP). DRAFT draft-ietf-manet-cbrp-spec-01.txt, IETF, July 1999.

[9] C. Lucas. Self-organizing systems (sos) faq, November 2002.

[10] N. Mitton, A. Busson, and E. Fleury. Analysis of the self-organization in wireless multi-hops networks. Research Report RR-5328, INRIA, 2004.

[11] N. Mitton, A. Busson, and E. Fleury. Self-organization in large scale ad hoc networks. In *Third Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, Bodrum, Turkey, june 2004.

[12] N. Mitton, E. Fleury, I. Guerin-Lassous, and S. Tixeuil. Self-stabilization in self-organized multi-hops wireless networks. Research Report RR-5426, INRIA, December 2004.

[13] S. Ni, Y. Tseng, Y. Chen, and J. Sheu. The broadcast storm problem in a mobile ad hoc network. In *ACM/IEEE MobiCom*, pages 151–162, Seattle, WA, USA, august 1999.

[14] N. Nikaein, H. Labiod, and C. Bonnet. Ddr-distributed dynamic routing algorithm for mobile ad hoc networks. In *1st ACM international symposium on mobile ad hoc routing and computing*, Boston, MA, USA, November, 20th 2000. ACM.

[15] M. Pearlman, Z. Haas, and S. Mir. Using routing zones to support route maintenance in ad hoc networks. In *Wireless Communications and Networking Conference (WCNC 2000)*, pages 1280–1285. IEEE, September 2000.

[16] A. Qayyum, L. Viennot, and A. Laouiti. Multipoint relaying for flooding broadcast messages in mobile wireless networks. In *35th Annual Hawaii International Conference on System Sciences (HICSS'02)*, Big Island, Hawaï, January 2002. IEEE.

[17] R. Ramanathan and M. Steenstrup. Hierarchically-organized, multihop mobile wireless networks for quality-of-service support. *Mobile networks and applications*, 3:101–119, June 1998.

[18] I. Stojmenovic, M. Seddigh, and J. Zunic. Dominating sets and neighbor elimination-based broadcasting algortithms in wireless networks. *IEEE transactions on parallel and distributed systems*, 13(1), January 2002.

[19] D. Stoyan, S. Kendall, and J. Mecke. *Stochastic geometry and its applications, second edition*. John Wiley & Sons, 1995.

[20] K. Tchoumatchenko. *Modélisation de réseaux de communication par la géométrie stochastique*. PhD thesis, ENS, Paris, Décembre 1999.

[21] J. Wu and H. Li. A dominating-set-based routing scheme in ad hoc wireless networks. *special issue on Wireless Networks in the Telecommunication Systems Journal*, 3:63–84, 2001.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifi que
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