

Hector is an Energy efficient Tree-based Optimized Routing protocol for wireless networks

Nathalie Mitton¹, Tahiry Razafindralambo¹, David Simplot-Ryl¹, Ivan Stojmenovic^{2,3}

¹CNRS/INRIA/Univ. Lille 1, France - {firstname.lastname}@inria.fr

²SITE, University of Ottawa, Canada - ivan@site.uottawa.ca

³ECE, University of Birmingham, UK

Abstract—This paper considers the problem of designing power efficient routing with guaranteed delivery for sensor networks with known distances between neighbors but unknown geographic locations. We propose HECTOR, a hybrid energy efficient tree-based optimized routing protocol, based on two sets of virtual coordinates. One set is based on rooted tree coordinates, and the other is based on hop distances toward several landmarks. In our algorithm, the node currently holding the packet forwards it to its neighbor that optimizes ratio of power cost over distance progress with landmark coordinates, among nodes that reduce landmark coordinates and do not increase tree coordinates. If such a node does not exist then forwarding is made to the neighbor that reduces tree based distance and optimizes power cost over tree distance progress ratio. Our simulations show the superiority of our algorithm over existing alternatives while guaranteeing delivery, and only up to 30 % additional power compared to centralized shortest weighted path algorithm.

I. INTRODUCTION

Wireless ad hoc networks, especially sensor networks, have received a lot of attentions in recent years due to their potential applications in various areas such as monitoring, security and data gathering. However, they have some limitations compared to wired infrastructure networks. Energy consumption and scalability are two challenging issues when designing sensor network protocols such as routing protocols since they operate on limited capacity batteries while the number of deployed sensors could be very large.

Position awareness in sensor networks improves the efficiency of route discovery and broadcasting algorithms. The fundamental idea behind position awareness (referred also as *geographic* or *geometric* information) is to provide a global position information to each node in the network. This information can be obtained through devices such as GPS or Galileo. Protocols using geographic information for routing (MFR [29], Cost-over-Progress [27], GFG [15]) are competitive alternatives to the classical routing protocols for wireless ad hoc networks (DSR [16], AODV [24], OLSR [7]). Indeed, classical routing protocols exchange $O(n^2)$ messages for route discovery and require $O(n)$ routing states at each node where n is the total number of nodes. On the other hand, in geographic routing protocols, nodes only need to store their and their neighbor's coordinates.

Nevertheless, position information provided by devices is not always a feasible solution for sensor networks since GPS do not work in every environments. GPS are bulky, energy-costly and expensive. Without such positioning devices, the option is to assign nodes 'virtual' geographical coordinates with an *internal location service*. These virtual coordinates do not necessarily embed global positioning information. They just have to be consistent to allow routing. Internal location services have already been studied in the literature. The first

common approach proposed in VCap [5], JUMPS [3], VCost [11], Gliders [13] or APS [20], [22] consists in computing a distance based on node hop count from a set of landmarks to obtain a virtual position. This approach is easy to implement and performances are interesting in terms of stretch factor and energy efficiency for some of the algorithms cited above [11]. However, packet delivery is not guaranteed even if a route between the source and the destination exists. Indeed, several nodes may hold the same virtual coordinates and label unicity is required for guaranteeing delivery.

The authors of [6] propose an alternative approach. In LTP [6], labels are assigned to nodes by building a tree through a depth-first search on the network. Each node is assigned a label depending on its position in the tree. The routing paths are embedded in the labels. LTP guarantees the delivery but is not energy aware and may provide paths with a high stretch factor.

In this paper, we focus on designing an energy-aware and scalable routing protocol which guarantees delivery for sensor networks where nodes are not aware of any positioning information. We introduce HECTOR, a Hybrid Energy-efficient Tree-based Optimized Routing protocol. HECTOR builds two sets of virtual coordinates: (i) virtual coordinates similar to the ones built in VCost, *i.e.* based on a node hop count distances to landmarks and (ii) a set of labels like in LTP. The first set of virtual coordinates allows HECTOR to find a greedy path in the forwarding direction of the destination. The second set of labels prevents HECTOR from reaching a dead end and the routing from failing. Based on these two sets of coordinates, a node holding a packet chooses its neighbor to forward the message in a Cost-over-Progress (COP [27]) fashion to save energy. The COP looks for nodes in the forwarding direction (here based on virtual coordinates or/and labels) and selects the one which minimizes the cost of transmission to this node over the progress made towards the destination.

HECTOR has the following properties : i) **Scalable**: Except the labeling steps which occurs at the bootstrap, to make a routing decision, a node has to be aware only of the location of itself, of its neighbors and of the final destination. Moreover, HECTOR is memoryless: no routing information has to be stored at the node where constant amount of information is embedded in the message. ii) **Loop free**: HECTOR is loop-free since it is a greedy routing which always makes any sender node s on the path forward to a node closer to the destination (in our coordinate system) than the sender node. iii) **Guaranteed delivery**: HECTOR guarantees the delivery thanks to its set of labels derived from a tree. In the worst case, HECTOR follows that tree which provides exactly one path between any pair of nodes. iv) **Energy efficient**: HECTOR selects the node which minimizes the cost over the progress towards the destination. Simulations show its superiority over existing alternatives while guaranteeing delivery, and only up to 30% additional power compared to centralized shortest weighted path algorithm.

Simulations show that HECTOR provides fair performances regard-

⁰This work was partially supported by NSERC Strategic grants STPGP 336406-07, the FP7 REGPOT-2007-3-01 ProSense grant, a grant from INRIA research action CARMA and the UK Royal Society Wolfson Research Merit Award

	Exact Position	Virtual Position
hop count (HC)	greedy [14], MFR [29]	VCap [5]
Energy-efficient (EE)	COP [27]	VCost [11]
HC+Guaranteed-delivery (GD)	GFG [15]	LTP [6]
EE+GD	EtE [10]	HECTOR

TABLE I
CLASSIFICATION OF GEOROUTING PROTOCOLS

ing the energy efficiency and the path length. In addition, as far as we know, it is the first algorithm to propose a geographic routing protocol where nodes are not aware of their positions, which is both energy-efficient and guaranteed-delivery. Moreover, HECTOR does not rely on specific assumptions (e.g. Unit Disk Graph) or any radio propagation model. It may be applied in any general topology. For all these reasons, to our knowledge, HECTOR has no competing solutions. Indeed, classical routing protocols such as AODV [24] or DSR [16] trigger a flooding from each source while HECTOR provides a fixed amount of flooding (only at bootstrap) from all landmarks and tree root. Existing geographical protocols either need positioning system such as MFR [29] or GFG [15] or do not guarantee delivery such as VCap [5] or VCost [11], or are not energy-aware like LTP [6].

This paper is organized as follows. We briefly cover related work in Section II. In Section III, we present the way of assigning the two sets of coordinates and introduce our model and assumptions. We describe HECTOR in Section IV. Then, we compare HECTOR's performances to existing methods in Section V by simulations and conclude in Section VI.

II. RELATED WORK AND MOTIVATIONS

Routing in wireless sensor networks is a challenging task. Many different approaches have been proposed in the literature. We can identify three main classes of routing protocols: (i) proactive routing such as OLSR [7] (ii) reactive routing such as AODV [24] and (iii) geographic routing, or georouting. This latter approach is receiving more and more attention since it is a memory-less and scalable approach, unlike the two other ones. In a geographic approach, every node is aware of the exact or virtual coordinates (position) of itself, its neighbors and of the destination. Exact location coordinates may be derived thanks to GPS [2] or any other position mean [3], [20].

Each of two families of georouting protocols (with exact and virtual coordinates) can be divided based on its properties with respect to the metric used (hop count or power), and whether or not it guarantees delivery. Therefore there are four classes of algorithms: (i) simple hop count based algorithms without guaranteed delivery, (ii) hop count based with guaranteed delivery, (iii) energy-efficient without guaranteed delivery or (iv) guaranteed-delivery and energy-efficient. Table I sums up the different categories and algorithms.

There are two well-known algorithms for the case where nodes are aware of their exact geographical coordinates thanks to GPS [2] or Galileo [1] or any estimation of them [12], [4]. In *Most Forward Routing with progress (MFR)* [29], the node S currently holding the packet for destination D forwards it to neighbor A whose projection on line SD is closest to D . In *greedy* routing [14], S forwards the message to the node that is closest to D . These are simple localized algorithms that do not guarantee delivery. A packet can be trapped in a local minimum and the algorithms fail to find a path to the destination leading to low delivery rates. In dense networks the algorithms perform well.

Greedy georouting has then been enhanced in two directions, toward changing hop count to other metric, and toward providing guaranteed delivery. Power aware greedy routing algorithms were

first studied in [28]. Instead of counting hops, power consumption on edges on a route was considered as the cost. An algorithm with general cost metric was proposed in [27]. Cost over Progress based routing [27] (COP) is a localized metric aware greedy routing scheme. A node forwards a packet to the neighbor closer to destination D such that the ratio of the energy consumed to the progress made (measured as the reduction in distance to D) is minimized. Though cost efficient, this algorithm does not guarantee delivery. Cost could be an arbitrary metric, such as hop count, power consumption, reluctance to forward packet, delay etc.

In [15], greedy routing is applied till reaching either the destination or a dead end. In latter case, Face routing is applied to recover from failure. Face routing requires the network topology to be a planar graph (i.e., no edges intersect each other). The graph planarization (through a Gabriel Graph [23] or a Relative Neighborhood Graph [30]) divides the graph in faces. The face that contains the line (SD), where S is the failure node, and D is the destination node, is traversed by right/left-hand rule (placing a virtual hand on the wall of the face) until a node A closer to destination than S is encountered. It has been shown in [15] that Face routing guarantees recovery traversing the first face. Greedy routing continues from A until delivery or another failure node is encountered. GFG guarantees delivery but uses hop count as metric, and is therefore not energy-aware.

EtE [10] combines advantages of greedy, power aware and guaranteed delivery approaches to give birth to an energy-efficient geographical routing with guaranteed delivery, for the case of using exact position information. EtE is based on a GFG routing in which both steps (greedy and Face) are energy aware. The algorithm applies Cost over Progress routing (with power as the cost) in both greedy and Face modes. For Face routing, the connected dominating set (CDS) is first constructed, and every edge on Gabriel graph over CDS, traversed by Face routing, is followed by Cost over Progress routing. The algorithms described so far rely on exact position information.

We now describe approaches that rely exclusively on virtual coordinates, derived from either relative distances or hop counting to a set of landmark nodes in the network, without the intervention of external location services. The general idea is to define a virtual coordinate system and use it to induce a routing protocol based on the virtual coordinates. We survey some of them below (VCap [5], Jumps [3], Gliders [13] or VCost [11]). A system of virtual coordinates based on three *landmarks* is proposed. Nodes are assigned a tuple of coordinates given as the number of hops the node is distant from each *landmark*. This virtual coordinate system establishment is described in detail in Section III-B.

In VCap [5] and JUMPS [3], nodes apply a greedy routing [14], based on the Hamming distance computed on these coordinates (instead of the Euclidean distance). The storage overhead for each sensor is limited to the storage of its coordinates and the coordinates of its neighbors. The authors show how the coordinate system is consistent for a given density of the network, i.e., nodes with the same coordinates lie within a limited number of hops from each other. A different approach is used in [13] where landmarks are selected more carefully after partitioning the nodes into tiles, and elaborate gradient descent procedures are used to route packets, and high communication and storage overhead is required to increase the delivery rate. However these approaches are neither energy-efficient nor guaranteed-delivery. Therefore VCost [11] proposes to use this system by applying a greedy cost-over-progress routing, like in COP [27], still based on the Hamming distance. VCost is energy aware but still does not guarantee delivery.

Liu and Abu-Ghazaleh [18] observed that increasing the number of landmarks cannot eliminate virtual anomalies since some portions of the network may be 1-connected to the rest of network. They propose a one-dimensional virtual coordinate system based on depth first search (DFS) preorder traversal of the graph. Starting from a root node, nodes are labeled 1, 2, 3,... with label assigned when a node is visited for the first time. Each node m also has an interval $[m, q]$ starting from itself until all its children nodes are assigned, before traversal returns back to its parent. Routing is based on these labels. Current node may have few forwarding options; each of them is a neighbor that contains destination label within its interval of labels. Forwarding to a child node is favored to forwarding to parent node.

In LTP [6], the authors introduce a new coordinate system, based on a tree construction. Each node is assigned a label which embeds the path between this node to any other node in the network, based on the path in the tree which is unique. The labeling process of LTP is described in more details in Section III-A. Thanks to this labeling, LTP ensures the delivery of the message and the success of the routing but is not energy aware and may provide paths which are much longer than the optimal one.

In this paper, we propose a routing protocol which combines early results from the literature in order to provide a protocol routing which is at the same time (i) energy efficient, (ii) guaranteed delivery and (iii) does not need any external position information.

III. PRELIMINARIES

Our routing process uses two sets of coordinates (V, T) . $V(u)$ is the set of coordinates of node u used to provide a progress in the geographic graph, thus limiting the stretch factor of the path length, but which cannot ensure the delivery if used alone. We use V coordinates based on landmark hop distances, like in VCost [11]. $T(u)$ is the set of labels that allows to guarantee packet delivery, i.e. if the network is connected, T coordinates provide a path between any pair of nodes. We use T coordinates like in LTP [6]. Each of these coordinates is associated to a distance: d_V and d_T respectively in order to measure a progress over each kind of coordinates. In the rest of this paper we will refer as 'virtual coordinates' for V coordinates and to 'labels' for T coordinates.

A. Building T labels

We build T labels in the same fashion as in LTP [6]. This labeling is performed through a tree construction. The tree is built iteratively from the root to the leaves. At bootstrap, a node is designed as root. This node may be a special node such as a fixed landmark. At each step, every freshly labeled node queries its unlabeled neighbors and then gives a label to each answering node. If $l(u)$ is the label of node u , the k^{th} neighbor of node u is labeled $l(u)k$. Figure 1(a) gives an example of how the nodes are labeled. We assume here that the root is node 4. The tree gives the shortest path in number of hops from the root to any other node.

The distance used in the tree is based on label size and common prefix which can give the hop distance between any two nodes of the network. Thus the distance between node a and node b is $d_T(a, b) = |l(a) - l(c)| + |l(c) - l(b)|$ where c is the lowest common ancestor of a and b and $l(a)$ is the label size of node a . From Figure 1(a) the distance between node 9 and node 5 is thus $d_T(9, 5) = |l(9) - l(4)| + |l(4) - l(5)| = |3 - 1| + |1 - 2| = 3$.

By building several trees, we can have a vector of labels, with one label per node for each tree. We finally have $T(u) = \{l_i(u)\}_{i=0, \dots, t-1}$ where t is the number of trees. If we have only one tree, $T(u) = \{l(u)\}$.

As described in [6], the path is encoded in the labels. There exists a path encoded in node labels between any two nodes of the network. This path is the path in the tree, which, by definition, always exists and is unique (for $t = 1$).

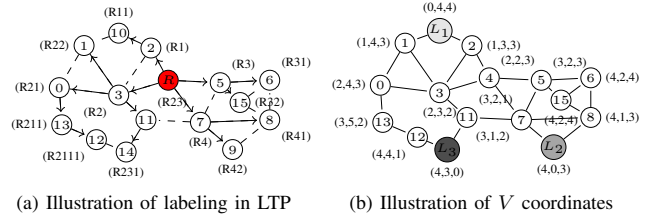


Fig. 1. In Fig. 1(a) The tree root is node 4 and has the label R . Node 13 is labeled $R211$ since it is the first child of node 0 which has label $R21$. Dashed lines represent physical links. In Fig. 1(b) Node 4 has coordinates $(2, 2, 3)$ since it is 2-hop away from landmarks 1 and 2 and 3-hop away from landmark 3.

B. Building V coordinates

These coordinates are similar to the ones in VCap [5], JUMPS [3] or VCost [11]. Several nodes, L_1, \dots, L_k with $k \geq 3$, in the network are distinguished as *landmarks*. Each landmark broadcasts a beacon in the network incremented at each hop. From it, an arbitrary node x knows its virtual coordinate vector $V(x) = (h_1, \dots, h_k)$ where h_i is the hop-distance between x and L_i . Figure 1(b) plots an example of how nodes are assigned virtual coordinates. We suppose 3 landmarks: nodes 10, 9 and 14. Every node thus has a 3-dimensional vector as coordinates constituted by the number of hops between itself and every landmark. For instance, node 0 can reach Landmark 1 (node 10) in 2 hops, Landmarks 2 (node 9) in 4 hops and Landmark 3 (node 14) in 3 hops. Its virtual coordinate is thus $V(0) = (2, 4, 3)$.

The distance used on these virtual coordinates is d_V where $d_V(u, u')$ is the Hamming distance from node u to node u' on V coordinates ($d_V(u, u') = \sum_{i=1}^M |h_i(u) - h_i(u')|$). For example, on Figure 1(b), the distance $d_V(0, 8)$ between node 0 ($V(0) = (2, 4, 3)$) and node 8 ($V(8) = (4, 1, 3)$) is $d_V(0, 8) = |4 - 2| + |1 - 4| + |3 - 3| = 2 + 3 + 0 = 5$.

Obviously, using only these coordinates does not guarantee delivery since the node coordinates are not unique (i.e. several nodes may have the same virtual coordinates) and thus do not identify a single node. This is for example the case for nodes 6 and 15 on Figure 1(b) which are both labeled with $(4, 2, 4)$.

C. Assumptions and Notations

Let $N(u)$ be the set of physical neighbors of node u , i.e. the set of nodes in communication range of node u . Let $\delta(u)$ be the cardinality of this set, also called the degree of node u : $\delta(u) = |N(u)|$. We define $N_V(u, u')$ as the set of neighbors of node u which reduce the distance to node u' , regarding the V coordinates: $N_V(u, u') = \{v | v \in N(u), \text{ and } d_V(v, u') < d_V(u, u')\}$. Similarly, $N_T(u, u')$ is the set of neighbors of node u which reduce the distance to node u' , in T coordinates: $N_T(u, u') = \{v | v \in N(u), d_T(v, u') < d_T(u, u')\}$.

The most common energy model [26] is as follows: $cost(r) = r^\alpha + c$ if $r \neq 0$, 0 otherwise, where r is the distance separating two neighboring nodes; c is the overhead due to signal processing; α is a real constant (> 1) that represents the signal attenuation. The optimal transmission radius, r^* , that minimizes the total power consumption for a routing task is equal to: $r^* = \sqrt[\alpha]{\frac{c}{\alpha - 1}}$ assuming that nodes can be placed on a line toward the destination [28].

Let's introduce the functions COP_T and COP_V as functions defining selection criteria of s 's next hop toward d in a cost-over-progress fashion [27] over coordinates T and V respectively. s selects node b which minimizes COP_T or COP_V as defined later in Section 1. These functions are as follows: $COP_T(u, v, d) = \frac{cost(|uv|)}{d_T(u, d) - d_T(v, d)}$ and $COP_V(u, v, d) = \frac{cost(|uv|)}{d_V(u, d) - d_V(v, d)}$, where $|uv|$ is the Euclidean distance between nodes u and v .

In this paper, we assume every node is able to control its transmitting power (and thus its range) and to estimate the Euclidean distance between itself and every of its neighbor, based on the received signal strength (RSSI) [20], or time difference [25] or angle of arrival [21].

IV. HECTOR

A. Algorithm description

Each node u has two sets of coordinates (V, T) as defined in Section III. For the sake of clarity, we use only one tree for labeling. The algorithm can then be easily extended to k trees.

The routing algorithm combines advantages of both kinds of coordinates : (i) virtual coordinates like in VCost [11] allow to minimize the path length and (ii) labels like in LTP [6] allow to avoid reaching a dead end and to guarantee delivery.

The basic idea is the following. A source node s holding a packet for a destination node d performs an energy-efficient greedy routing scheme in a VCost fashion. In order to avoid to be trapped in a local minima, the routing algorithm selects the next hop with regards to not only the virtual coordinates but also regarding the labels. The routing process runs as follows. When node u receives a message for node d , it first considers its neighbors in the forward direction, based on both their labels and virtual coordinates. It only considers nodes v for which d_T distance toward d is equal or smaller than the tree distance between u and d ($d_T(v, d) \leq d_T(u, d)$). Such neighbors always exist (whenever source and destination nodes are connected) because of convergence of label based routing. The algorithm first checks whether any one of these nodes also provides a progress with respect to landmark coordinates. Let $H = N_T(u) \cap \{N_V(u) \cup v \mid d_T(v, d) = d_T(u, d)\}$ be the set of such nodes.

If $H \neq \emptyset$ then u selects its next hop among the nodes in H (thus reducing the distance toward the destination regarding coordinates V and not increasing distance regarding T labels) as the node v which provides the best ratio cost over progress to the destination regarding the virtual coordinates (v such that $COP_V(u, v, d) = \min_{w \in N_V(u)} COP_V(u, w, d)$).

Otherwise (that is, if $H = \emptyset$), node selects its neighbor v which provides the best ratio cost over progress (like in [27]) to the destination regarding the labels (v such that $COP_T(u, v, d) = \min_{w \in N_T(u)} COP_T(u, w, d)$). Such a node always exists since there always exists exactly one path in the tree between any two nodes. In case of ties, the next hop is chosen at random between candidates.

Algorithm 1 formally describes this routing process.

B. Algorithm quality

Lemma 1: A packet cannot transit from a node u to another node v if $V(u) = V(v)$ (or if $d_V(u, d) < d_V(v, d)$) unless there is an absolute progress regarding T labels (if $d_T(u, d) > d_T(v, d)$).

Proof: Let us assume that node u holds a packet for a destination d . Suppose that nodes u and v have the same V coordinates ($V(u) = V(v)$) or that v is farther than node u regarding V coordinates ($d_V(u, d) < d_V(v, d)$). Then $v \notin N_V(u, d)$ which means that $v \notin H$. The selected next hop thus belongs to $H' = \{v \mid COP_T(u, v, d) = \min_{i \in N_T(u)} COP_T(u, i, d)\}$, which contains every neighbor of u closer to d than u regarding T labels. Thus, if

Algorithm 1 Run at each node u on the routing path toward d to select next hop.

```

1: if  $u = d$  then
2:   exit {*/Routing has succeeded*/}
3: else
4:    $H = \{N_T(u, d)\} \cup \{v \mid d_T(v, d) = d_T(u, d)\} \cap \{N_V(u, d)\}$ 
5:   if ( $H = \emptyset$ ) then
6:     {*/No node is closer to  $d$  than  $u$  on both  $V$  and  $T$ .*/}
7:      $H' = \{v \mid COP_T(u, v, d) = \min_{w \in N_T(u)} COP_T(u, w, d)\}$ 
8:   else
9:      $H' = \{v \mid COP_V(u, v, d) = \min_{w \in H} COP_V(u, w, d)\}$ 
10:  end if
11:  if ( $|H'| > 1$ ) then
12:     $Next\_Hop = rand(H')$ 
13:  else
14:     $Next\_Hop = v$  where  $H' = \{v\}$ 
15:  end if
16: end if

```

node v is chosen as the next hop, that means that $v \in H'$ and thus provides a progress regarding T labels. Note that in the worst case *i.e.* when the progress on T labels is minimal, the next hop is either the parent or a child of node u . ■

Lemma 2: The routing protocol described in Alg. 1 is loop free.

Proof: We introduce an order among all nodes with respect to combined distance to destination d . Consider $d_T(u, d)$ as the primary key, and $d_V(u, d)$ as the secondary one. Two nodes are sorted by their primary key. In case of ties, the secondary key is used. Thus $u < v$ if and only if $d_T(u, d) < d_T(v, d)$ or ($d_T(u, d) = d_T(v, d)$ and $d_V(u, d) < d_V(v, d)$). Let us assume that node u_0 is the source of a packet, d its destination and node u_1 the next hop chosen by node u_0 . If $H \neq \emptyset$ then $d_T(u_1, d) \leq d_T(u_0, d)$ as per restriction. Also similarly $d_V(u_1, d) < d_V(u_0, d)$. Therefore $u_1 < u_0$ in our order. Let $H = \emptyset$. Then $d_T(u_1, d) < d_T(u_0, d)$ and therefore again $u_1 < u_0$. This means that $u_1 < u_0$ based on H and H' construction. Our routing process therefore strictly reduces distances to destination in the order defined by given primary and secondary keys. This means that loops cannot be created. ■

Lemma 3: In the routing protocol described in Algorithm 1, there always exists a next hop that is closer to the destination regarding both sets of virtual coordinates.

Proof: Let us consider a source u and a destination d . By construction, if a node in $N_V(u, d)$ is chosen as the next hop, this ensures a progress on the V coordinates. If the next hop is chosen in $N_T(u, d)$ this ensures a progress in the tree toward the destination. The progress will occur since $N_T(u, d)$ is a nonempty set. ■

It is worth noting that the progress made on V is more important than the progress made on T labels in the geographical space. Indeed, the next hop in the T labels can have the same V coordinates and thus more or less the same Euclidean distance to the destination. These lemmas show that the routing protocol HECTOR described in Algorithm 1 always works in a greedy way. The greedy aspect provided by this algorithm makes it simple, memoryless and scalable.

Theorem 4: The routing algorithm described in Algorithm 1 guarantees delivery.

Proof: Each node has a unique label due to the labeling process described in Section III. This ensures that the destination of a packet is unique and that at each step of the routing protocol, a next hop closer to the destination can be found. Based on Lemma 1, Lemma 2 and Lemma 3 if a path exists (if the network is connected), the routing protocol will find it in a greedy way. ■

V. SIMULATION RESULTS

This section presents the simulation results of our algorithm. We compare our solution to the geographical algorithms of the literature which assume no position information: VCost [11], which is the best algorithm known regarding energy-efficiency and LTP [6], which is one of the first known to guarantee delivery. In order to further evaluate the energy saving contributions of HECTOR, we also compare it to its variant HECTOR' which selects the next hop as the node which maximizes the progress towards the destination (*i.e.* it considers that $cost(|uv|) = 1 \forall u, v$ and tries to minimize COP_T or COP_V). HECTOR' guarantees delivery but uses hop count as metric and is not energy aware. We first present the simulation setup and then give some performance results about energy consumption overhead, mean path length and mean hop length.

A. Simulation setup

As we focus our performance evaluation study on network layer mechanisms, for our performance results to be independent of the lower layers, we chose to use our home-made simulator that assumes an ideal MAC (no packet collision, no delay) and Physical layers (no interference, BER = 0). The network can be described as follows. Nodes are deployed in a 1000×1000 square following a two dimensional Poisson Point Process with different intensities λ . In such a Poisson Point Process, the total number of nodes is probabilistic and is obtained from a Poisson Law of intensity λ which is correlated to the mean node degree δ : $\lambda = \frac{\delta}{\pi R^2}$. Nodes are uniformly distributed over the area. Nodes can adapt their range between 0 and $R = 200$. We only consider connected networks.

We compare HECTOR, LTP [6] and VCost [11] for the same samples of node distribution, same source and destination pairs, both randomly chosen. Landmarks and the tree root are randomly chosen among the nodes. Finally, to show the impact of the use of the two sets of coordinates over the guaranteed delivery, we evaluate the performances of the routing schemes over a homogeneous network and over a topology with a crescent hole (see Fig.. 2).

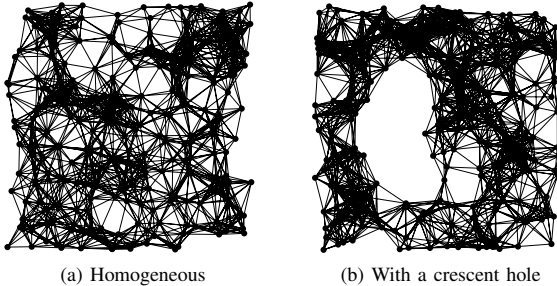


Fig. 2. Network topologies.

We evaluate the energy consumption overhead (ECO) of each algorithm based on the energy model described in Section III. As in [26], we use $c = 10^7$ and $\alpha = 4$, which leads to an optimal range of $r^* = 100$ [17]. To further evaluate the routing protocols, we computed their energy overhead using as reference the optimal centralized energy weighted shortest path (SP) (Dijkstra algorithm [9]). We let e_i and e^* be the energy consumed using any described protocol and the centralized SP protocol, respectively. We define the energy overhead as the ratio $\frac{e_i - e^*}{e^*}$. We also evaluate the mean path length and mean hop length obtained for each protocol and give visual results of routing process.

B. Results

Energy consumption overhead when VCost succeeds. Fig. 3 plots the ECO for paths provided by the different algorithms when VCost succeeds for a given source-destination pair. The energy overhead is plotted depending on the mean node degree and on the number of landmarks used to build V coordinates. We can see from this figure that HECTOR provides the lowest overhead within the protocols that guarantee delivery. We can see that the node degree and the number of landmarks have a limited impact on the performances of each protocol since the figure plots the energy overhead once a path has been found. Since the environment is homogeneous, the impact of these parameters is thus negligible on the path features. Fig. 4 plots the same results in a topology with a hole, still when VCost succeeds. Here we can also see that the performances of HECTOR are the best within the protocols that guarantee delivery.

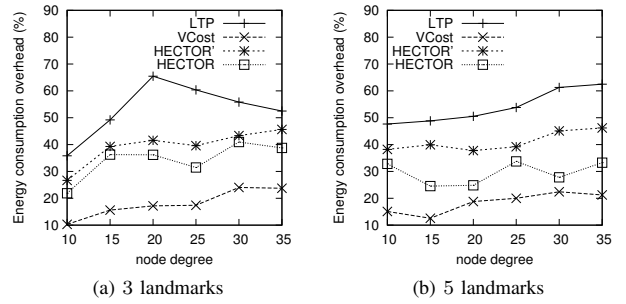


Fig. 3. ECO when VCost succeeds for a homogeneous topology.

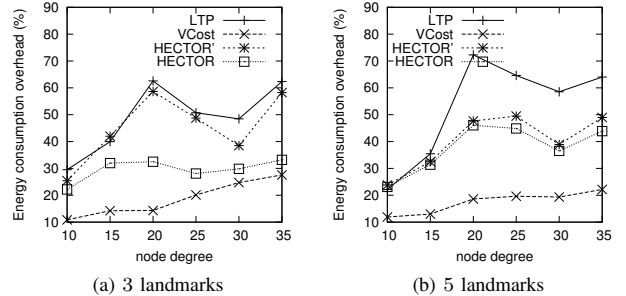


Fig. 4. ECO when VCost succeeds for a topology with a hole.

As expected, for each case, HECTOR provides a greater overhead than VCost. This is due to the routing process in HECTOR that tries to provide a progress in the tree at any step. Therefore, the tree root position is important for minimizing the energy consumption for a given source and destination but it is not possible to have an optimal tree root position for all possible source-destination pairs.

Nevertheless, as Fig. 5(a) shows, the success rate of VCost is far from 100% and is not the same following the different scenario. We can note that the more VCost succeeds, the more HECTOR is energy-efficient and sticks to VCost performances. This is because, in Algorithm 1, the V coordinates are chosen uppermost. On the other hand, the less VCost succeeds, the more HECTOR sticks to LTP performance. If VCost fails, that means that there is no path following V coordinates and thus HECTOR algorithm follows T labels to ensure packet delivery, like in LTP. This is also confirmed by results plotted by Fig. 5(b) which are the percentage of times

HECTOR progresses over V coordinates rather than only on T labels. This is correlated with the success rate of VCost which only follows V coordinates.

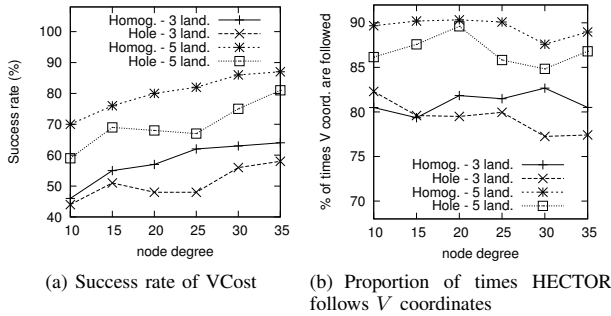


Fig. 5. Success rate of VCost routing algorithm (a) and number of times HECTOR follows V coordinates (b) for each scenario.

Energy consumption overhead when VCost fails. When VCost fails, HECTOR has to follow T labels to reach the destination. This feature is one of the main contributions of HECTOR and cannot be observed on Fig. 3 and 4 since the latter ones plot results for simulation runs where VCost succeeds. Therefore, Fig. 6 and 7 plot the results of HECTOR, HECTOR' and LTP for every simulation run, about paths on which VCost fails.

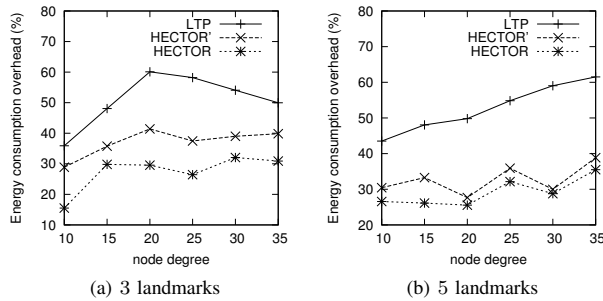


Fig. 6. ECO when VCost fails for a homogeneous topology.

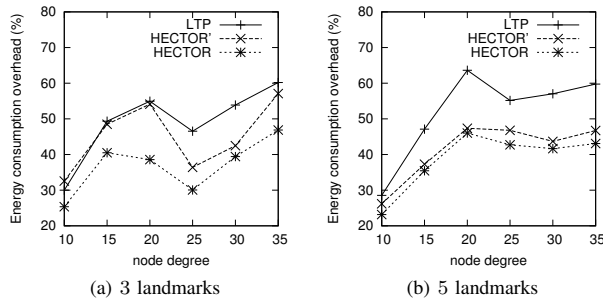


Fig. 7. ECO when VCost fails for a topology with a hole.

In Fig. 6 and 7, HECTOR is the algorithm that provides the best performances regarding energy consumption, followed by HECTOR'. LTP, once again, is the least performing algorithm. Moreover, as expected, we can note that the global behavior of HECTOR and HECTOR' is the same as LTP's. As already mentioned, this is because, when there is no progress over V coordinates, HECTOR

and HECTOR' follow the T labels, like LTP, and so on till reaching a node which can provide a progress regarding V coordinates.

Hop length. Fig. 8 plots the mean hop length along the routing path for every algorithm. The optimal hop length (based on energy consumption) is plotted as a reference. Results are similar for other choices of the number of landmarks and the topology. We can notice that VCost and HECTOR follow edges which lengths are close to the optimal one [28] in every case. The mean hop length is greater than the optimal one with HECTOR because the choice of the next hop is conditioned by the progress made over T labels, which leads to greater hop length because of tree construction. Indeed, because of the labeling process, close nodes are mainly at the same level in the tree and thus the progress providing by them is null. Therefore, since nodes try to minimize the cost over progress ratio, they generally try to maximize the progress and thus reaching nodes a little bit further.

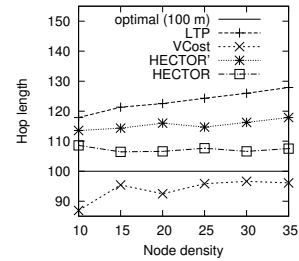


Fig. 8. Mean hop length.

Path Length. Fig. 9 plots the path length in number of hops when VCost succeeds. We can notice that VCost is the algorithm which provides the shortest paths. This is because it is the only algorithm to select the next hop in the forwarding direction at each hop. LTP is the one achieving the longest paths since it follows the path in the tree, sometimes with shortcuts between branches but which is rarely the shortest path. The tree construction affects the mean hop length of LTP. The impact of having an energy efficient tree or a tree with optimized range is left to future works. As already mentioned, HECTOR tries to stick to VCost when it is possible. HECTOR' acts as VCost but since it is not energy-aware, HECTOR' takes long edges and thus gets shorter paths than HECTOR. Note that in the worst cases when HECTOR can never provide a progress over V coordinates and that there is only one candidate that provides a progress over T labels, it also follows the tree. In this latter case, the path length provided by HECTOR is longer than the one achieved by LTP since they both follow more or less the same route but LTP takes longer edges and HECTOR tries to fit the optimal range.

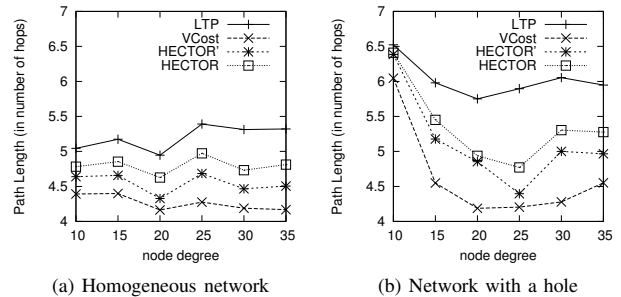


Fig. 9. Path length in number of hops when VCost succeeds. (3 landmarks)

When VCost fails, HECTOR and HECTOR' take decision based on T labels only and thus, HECTOR and HECTOR' stick to LTP. Thereby, HECTOR provides longer paths than LTP and HECTOR', which are not energy aware and takes long links while HECTOR favors edges less energy-costly. This behavior is highlighted in Fig. 10 which plots the path length in number of hops when VCost fails.

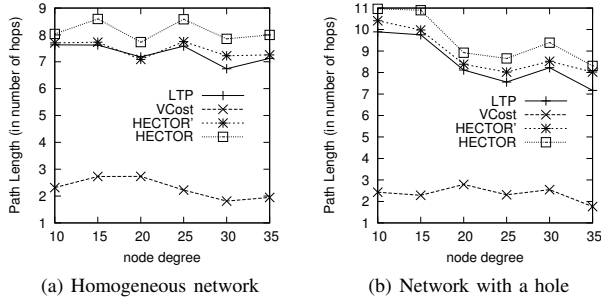


Fig. 10. Path length in number of hops when VCost does not succeed. For VCost : number of hops before failing. (3 landmarks)

Another interesting feature to point out is that globally, HECTOR provides longer paths than HECTOR' and VCost while it spends less energy. This also shows that HECTOR distributes the energy spending over the nodes on the paths.

Path shapes. Fig. 11 shows example of the paths followed by each protocol in a network with a crescent hole. Five landmarks are randomly chosen and the tree root is the red/black node in the middle of the network. Source and destination are also randomly chosen. These schemes clearly show the behavior of each algorithm.

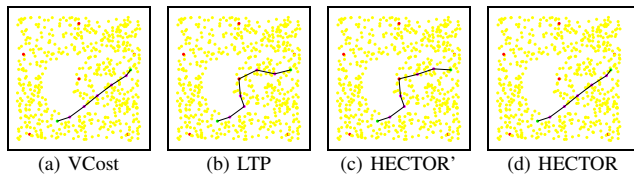


Fig. 11. Illustration of the paths followed by each algorithm with the use of 5 landmarks. Source is in the right side. VCost and HECTOR follow the same path while LTP and HECTOR' passes through the tree root.

We can see in Fig. 11 that VCost and HECTOR follow exactly the same path. This means that every hop provides a progress on both V and T coordinates. It is nevertheless worth noting that this would not appear in the general case. Indeed, for HECTOR to follow exactly the same path that VCost, a progress has to be made at each step on both sets of coordinates. Even if a progress is made on V coordinates by a node u , to be chosen, this node u has to also provide a progress on T labels, which is strongly related to the tree root, the source and the destination position. At the contrary, HECTOR' does not try to minimize the COP and thus the first hop is different than in VCost and directs it toward the hole. From it, in order to provide a progress regarding T labels and avoiding the dead end, HECTOR' has to follow the path in the tree, like in LTP. The path followed by LTP goes through the tree root which, in this particular case, increases the path length.

In Fig. 12, the same simulation is run with different source and destination pairs. The tree root is also modified. We can see from this figure that the path followed by VCost falls into a dead end after the second hop. One may think that when VCost fails, HECTOR follows

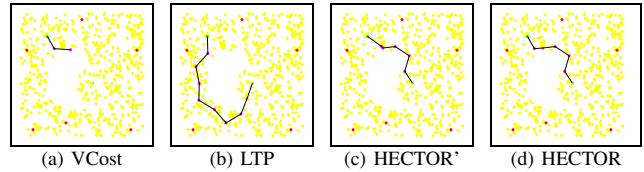


Fig. 12. Illustration of the paths followed by each algorithm with the use of 5 landmarks. VCost fails after the second hop, LTP passes through the tree root and HECTOR combines both T and V coordinates

the path of LTP. We can see on Fig. 12(d) that HECTOR first follows the V coordinates and then avoids the dead end encountered by VCost by using both T and V coordinates. This example shows how the combination of both T and V coordinates can guarantee delivery and optimize the path length. Once again, LTP follows the complete path in the tree and provides a very long path.

It is worth noting that the landmarks and the tree root positions have a great impact on the routing process. In VCost, landmarks position may affect the success rate. In LTP, the tree root position may increase the path length, and in HECTOR, the path may be different depending on these positions.

C. Enlarging the network

Till now, we have evaluated HECTOR by comparing it to other existing algorithms by running them in a restricted area and by making the node density grow. In this section, we fix the node density to $\delta = 15$ and the maximum node range radius to $R_{max} = 200$ and make the network area size expand.

Indeed, in such a scenario, the energy consumption will necessary grow since nodes may be further one from the others and more hops are needed to connect them than in previous scenarios. This section allows us to check the scalability of HECTOR in very large networks by being sure that we still ensure a low ECO.

Fig. 13 plots the ECO of the routes taken by each algorithm when the network size grows, for 5 landmarks. The results are similar for 3 landmarks. The abscissa axis plots the factor by which the network size has been multiplied. We only plot results where VCost fails since these runs represent the most energy costly results and the longest paths, as seen in previous sections.

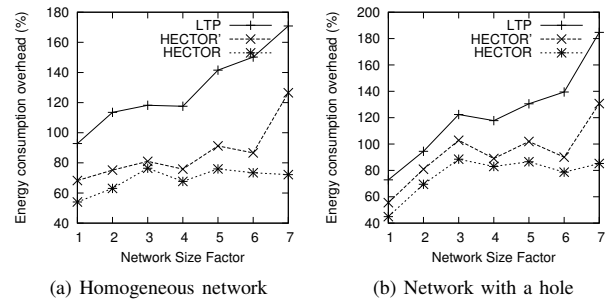


Fig. 13. ECO when the network grows for 5 landmarks.

Note that for homogeneous networks, even if the network grows as well as the route length, the energy overhead compared to the optimal shortest path consumed by HECTOR grows slowly with the network size. This is because HECTOR can follow V coordinates, and thus can have an energy consumption close to the optimal. Therefore, the energy consumed by paths followed by each algorithm is within a

constant factor of the optimum. Also note that for distributions with a hole, the energy overhead tends to increase with the network size. This is due to the fact that the bigger the network (and longer the routes), the more likely greedy routing encounters a dead end and thus HECTOR has to follow the T labels and the tree, which gives longer paths and thus consumes more energy.

VI. CONCLUSION

In this paper, we introduce HECTOR, a Hybrid Energy-efficient Tree-based Optimized Routing protocol. HECTOR is a geometric routing protocol designed for wireless sensor networks. Unlike the approaches proposed in the literature, HECTOR is *i)* based on virtual coordinates, *ii)* energy aware, *iii)* guarantees delivery, *iv)* scalable and *v)* do not assume any propagation radio model such as the Unit Disk Graph. These properties are provided by the combination of two sets of virtual coordinates used in HECTOR: landmark-based coordinates and tree-based coordinates. Simulation results show that HECTOR exhibits fair performances compared to the protocols presented in the literature, regarding energy consumption and stretch factor. Moreover, as far as we know, HECTOR is the first geographic routing protocol based on virtual coordinates that is both energy-efficient and with guaranteed delivery. Note that in this paper we use landmark-based coordinates for the energy efficient step but any other coordinate system may be used instead, included GPS localization. Therefore, we intend to explore another coordinate system than the landmark-based one in order to avoid the preprocessing flooding step by applying dominating set [8].

The next step of this work is to provide a more efficient and optimized way to build the virtual coordinates in HECTOR. Indeed, the weakness of HECTOR regarding energy consumption is due to the underlying tree used for one set of coordinates. Building a tree with energy-aware properties would make HECTOR even more efficient. Moreover, we also want to reduce the space needed to store the virtual coordinates especially for the labels such as in [18]. At last, another aspects to analyze are the study of HECTOR towards node mobility, asymmetric links and extension to heterogeneous networks [19].

REFERENCES

- [1] Galileo Positioning System. <http://www.aatl.net/publications/galileo.htm>.
- [2] GPS: Global Positioning System. www.gps.gov.
- [3] F. Benbadis, J.-J. Puig, M. Dias de Amorim, C. Chaudet, T. Friedman, and D. Simplot-Ryl. Jumps: Enhanced hop-count positioning in sensor networks using multiple coordinates. *Ad Hoc & Sensor Wireless Networks*, to appear.
- [4] S. Capkun, M. Hamdi, and J.P. Hubaux. GPS-free positioning in mobile ad-hoc networks. In *HICSS'01*, Hawaii, USA, 2001.
- [5] A. Caruso, S. Chessa, S. De, and A. Urpi. GPS-free coordinate assignment and routing in wireless sensor networks. In *INFOCOM'05*, pages 150–160, FL, USA, 2005.
- [6] E. Chávez, N. Mitton, and H. Tejada. Routing in wireless networks with position trees. In *Ad Hoc Now'07*, Mexico, 2007.
- [7] T. Clausen, P. Jacquet, A. Laouti, P. Muhlethaler, A. Qayyum, and L. Viennot. Optimized Link State Routing Protocol (OLSR), 2003. RFC 3626.
- [8] M. Couture, M. Barbeau, J. Bose, and E. Kranakis. Incremental construction of k -dominating sets in wireless sensor networks. *Ad Hoc & Sensor Wireless Networks*, 5:47–68, 2008.
- [9] E.W. Dijkstra. Solution of a problem in concurrent programming control. *Communications of the ACM*, 8(9), 1965.
- [10] E. H. Elhafsi, N. Mitton, and D. Simplot-Ryl. End-to-End Energy Efficient Geographic Path Discovery With Guaranteed Delivery in Ad hoc and Sensor Networks. In *PIMRC*, France, 2008.
- [11] E.H. Elhafsi, N. Mitton, and D. Simplot-Ryl. Cost over progress based energy efficient routing over virtual coordinates in wireless sensor networks. In *2pWSN'07*, Helsinki, Finland, 2007.
- [12] E. Ermel, A. Fladenmuller, G. Pujolle, and A. Cotton. On selecting nodes to improve estimated positions. In *IFIP, Conference on Mobile and Wireless Communication Networks (MWCN)*, France, 2004.
- [13] Q. Fang, J. Gao, L.J. Guibas, V. Silva, and Z. Li. GLIDER: gradient landmark-based distributed routing for sensor networks. In *INFOCOM'05*, pages 339–350, FL, USA, 2005.
- [14] G.G. Finn. Routing and addressing problems in large metropolitan-scale internetworks. Technical Report ISI/RR-87-180, Information Sciences Institute (ISI), 1987.
- [15] H. Frey and I. Stojmenovic. On delivery guarantees of face and combined greedy-face routing in ad hoc and sensor networks. In *MOBICOM'06*, CA, USA, 2006.
- [16] D. B. Johnson, D. A. Maltz, and J. Broch. DSR: the dynamic source routing protocol for multihop wireless ad hoc networks. *Ad hoc networking*, pages 139–172, 2001.
- [17] J. Kuruvila, A. Nayak, and I. Stojmenovic. Progress and location based localized power aware routing for sensor networks. *Intl. Journal of Distributed Sensor Networks (IJDSN)*, 2:147–159, 2006.
- [18] K. Liu and N. Abu-Ghazaleh. Stateless and delivery guaranteed geometric routing on virtual coordinate system. *Ad Hoc & Sensor Wireless Networks*, to appear.
- [19] L. Mamatas and V. Tsaoussidis. Exploiting energy-saving potential in heterogeneous networks. *Int. J. Parallel, Emergent & Distributed Systems*, 23:309–324, 2008.
- [20] D. Niculescu and B. Nath. Ad hoc positioning system (APS). In *GLOBECOM'01*, TX, USA, 2001.
- [21] D. Niculescu and B. Nath. Ad hoc positioning system (APS) using AOA. In *INFOCOM'03*, CA, USA, 2003.
- [22] D. Niculescu and B. Nath. DV-based positioning in ad hoc networks. *Journal of Telecommunication Systems*, 22(1-4):267–280, 2003.
- [23] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. In *DIAL-M*, pages 48–55, WA, USA, 1999.
- [24] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-demand Distance Vector Routing, 2003. RFC 3561.
- [25] N.B. Priyantha, A.K.L. Miu, H. Balakrishnan, and S. Teller. The cricket compass for context-aware mobile applications. In *MOBICOM'01*, Italy, 2001.
- [26] V. Rodoplu and T. Meng. Minimizing energy mobile wireless networks. *IEEE Journal on Selected Areas*, 17(8):1333–1347, 1999.
- [27] I. Stojmenovic. Localized network layer protocols in sensor networks based on optimizing cost over progress ratio. *IEEE Networks*, 20:21–27, 2006.
- [28] I. Stojmenovic and X. Lin. Power-aware localized routing in wireless networks. *IEEE Trans. Paral. Distrib. Syst.*, 12(11):1122–1133, 2001.
- [29] H. Takagi and L. Kleinrock. Optimal transmission ranges for randomly distributed packet radio terminals. *IEEE Transaction on Communications*, com-22(3), 1984.
- [30] G. Toussaint. The relative neighborhood graph of a finite planar set. *Pattern Recognition*, 12(4):261–268, 1980.