GEOROY: A Location-Aware Enhancement to Viceroy Peer-to-Peer Algorithm

Laura Galluccio, Giacomo Morabito, Sergio Palazzo,
Marco Pellegrini, M. Elena Renda, Paolo Santi

Abstract

The success of experiences such as Seattle and Houston Wireless has attracted the attention on the so called *wireless mesh community networks*. These are wireless multihop networks spontaneously deployed by users willing to share communication resources. Due to the *community* spirit characterizing such networks, it is likely that users will be willing to share other resources besides communication resources, such as data, images, music, movies, disk quotas for distributed backup, and so on. To support resource exchange in these wireless mesh community networks, algorithms for efficient retrieval of information are required. In this paper we introduce *Georoy*, an algorithm for the efficient retrieval of the information on resource location based on the *Viceroy* peer-to-peer algorithm. Differently from Viceroy, Georoy exploits the capability of setting and managing a direct mapping between the resource ID and the node which maintains information about its location so as to speed up the search process. Simulation results show that Georoy enables efficient and scalable search of resources and can be successfully used in wireless mesh community networks.

Key words: Wireless mesh networks, community networks, resource localization, distributed hash tables, scalability.

A Introduction

Wireless mesh networks are a promising area for the deployment of new wireless communication and networking technologies [4,5].

One of the possible application scenarios for wireless mesh networks is the realization of *wireless community networks*, which are becoming increasingly popular since the advent of cheap wireless technologies such as IEEE 802.11.

Given the *community* spirit of such networks, it is expected that users will be willing to share also non-communication resources, such as data, images, music, movies, disk quotas for distributed backup, etc. It is therefore likely that peer-to-peer applications will play a fundamental role in enriching the services offered by community networks.

In this paper we consider one of the major problems to be solved in peer-to-peer applications, i.e., efficiently finding the resources currently available in the network, in the context of wireless mesh community networks. To address this problem, we propose a methodology for resource search in the network which exploits the feature of appropriately mapping the resource ID to the location of the node in the network which possesses information about the location of this resource. This feature can be exploited for performing geographic forwarding of requests and, thus, speed up traditional distributed hash table (DHT) algorithms. Our algorithm, denoted as *Georoy*, is a location-aware enhancement to Viceroy proposed in [12]. To the best of our knowledge, Viceroy is the only DHT algorithm proposed in the literature which provably ensures a good balance of the load generated by search requests among the peers composing the network. Since load balancing is essential to guarantee adequate performances in wireless mesh networks, we believe Viceroy is a good starting point for implementing efficient and scalable resource sharing in this scenario.

The emphasis in our design is on *scalability*, since we believe this will be a fundamental property of any solution tailored to wireless community networks. In fact, the coverage area of community networks is expected to increase up to an entire city area, and we envision that the number of network nodes composing the network will grow as well, up to hundreds or even thousands of nodes.

In the following sections we describe the Georoy algorithm and we formally prove that the set of logical links created by Georoy efficiently exploits the underlying physical wireless network. Finally, we verify through simulation that the search efficiency of Georoy (i.e., the average number of network-layer messages generated to satisfy a resource request) is as much as 7 times better than that achieved by Viceroy, and that it can be successfully used to perform efficient and scalable resource localization in wireless mesh community networks.

The rest of this paper is organized as follows. In Section B, we present related work, and we highlight the original contributions of our paper. In Section C we present the Georoy algorithm. In Section D, we describe the mobility management procedures for integrating Georoy into a realistic system, and in Section E we present a simulation-based performance evaluation of our peer-to-peer resource sharing platform. Finally, Section F concludes and discusses future research directions.

B Related work and basic idea

The problem of enabling efficient peer-to-peer (P2P) resource sharing has been widely studied in the literature, following the success of the Napster file sharing application [13]. Indeed Napster cannot be considered as a pure P2P approach, since the index of the files available in the network is maintained by a centralized server: when a new peer joins the network, it provides the catalog of the files it will to share to the centralized server, which handles also all the search requests issued by peers. Since the use of a centralized server creates a bottleneck (and unique point of failure) in the system, several later proposals adopted a distributed approach to solve the P2P resource sharing problem.

Gnutella v0.4 [1] is an example of flat, unstructured P2P network with no directory service: when a new peer joins Gnutella, it establishes a number of virtual links to other peers in the network according to a certain rule, thus forming an overlay network. When a peer receives (or issues) a search request, it first checks whether the request can be satisfied locally, otherwise it forwards the request to its neighbors in the overlay. The request is flooded in the network until its time to live (TTL) expires. Thanks to its fully distributed nature, Gnutella v0.4 displays better robustness than Napster, but the use of (limited) request flooding causes a considerable message overhead and reduces the accuracy of the search process.

In order to maintain the search efficiency provided by a directory service while not sacrificing robustness and scalability, a number of P2P approaches are based on a hierarchical organization of peers: network members are divided into a large number of peers that provide content (called *leaf peers* LPs), and into a smaller number of peers that implement local directory services and route search requests (called *super peers* SPs). In hierarchical P2P networks, each super peer provides a centralized directory service to a subset of the leaf peers. Leaf peers are connected to one or more super peers, to which they provide their catalog of shared resources. Super peers are interconnected by a number of virtual links, thus forming an overlay network at the super peer level. Search requests originate at leaf peers and are handled by super peers: a leaf peer sends its request to the super peer(s) to which it is connected to; if it cannot be satisfied locally, the request is flooded in the super peer overlay network until its TTL (Time To Live) expires. Examples of hierarchical P2P networks are Gnutella v0.6 [2] and KaZaA [3].

In the above described P2P approaches the virtual overlay network used for searching the available resources within the network is unstructured. A number of recent proposals is aimed at improving search accuracy and efficiency by imposing a certain desirable structure to the overlay network. This is the case of approaches based on the Distributed Hash Table (DHT) abstraction: a key is assigned to each resource available in the network (e.g., by hashing the resource name), and a node ID is associated with each peer (e.g., the IP address). A certain range of keys is assigned to each peer in the system, which is responsible for answering the queries in the range. Based on its ID, a peer establishes a limited number of virtual links with other peers in the network, forming a highly structured virtual overlay network. The search process can be seen as searching a key in a (distributed) hash table, and is performed by comparing the queried key with the node ID: if the key belongs to the range managed by the node, then the query is answered and returned to the originating peer, otherwise it is forwarded to a specific neighbor in the overlay, which is chosen depending on the key value. DHT approaches differ on the structure imposed to the virtual overlay and on the mechanism used to route search requests in the overlay. For instance, CAN [15] routes along a hypercube, Chord [21] routes along a ring, Viceroy [12] is based on a butterfly network, Tapestry [8] uses a tree, and Koorde a DeBruijn graph [9]. The interesting feature of DHT approaches is that queries can be correctly answered by traversing a limited number of links in the virtual overlay network (typically, $O(\log n)$ links, where n is the number of peers in the system). On the other hand, preserving this nice feature in presence of frequent peer join/leaves (as it is typically the case in P2P applications) is very challenging.

Typically, P2P systems are designed and optimized for use in wired networks (e.g., the Internet). A few recent works have considered the problem of designing and/or extending existing systems to work efficiently on wireless ad hoc networks. In these networks, many additional challenges must be faced when designing P2P applications. The main ones are related to the lack of a wired infrastructure, the unreliability of the wireless channel, and the node mobility. It has been observed that, in order to tackle these challenges, it is fundamental to carefully build the overlay network between the peers so that it closely matches the underlying physical network. In other words, it is desirable that a peer chooses as neighbors in the overlay, nodes that are not too far in the physical network[1]. This can be accomplished by using cross-layering, where the network layer and the P2P application exchange information in order to optimize system performance. An example of this approach is presented in [6], where Conti et al. present a cross-layer optimization of Gnutella[2]. Another approach is the ORION system proposed in [10], where overlay connections (which closely match the underlying network topology) are built on-demand and maintained only as long as necessary. Other work related to ours can be

[1] Similar concepts have been used also in traditional, Internet-based P2P designs. For instance, the approach presented in [16] builds the overlay by choosing neighbors depending on the link latency.
[2] The authors implement Gnutella v0.6, but they consider only super peers. So, their approach can be straightforwardly applied also to Gnutella v0.4.

find in [7,14,17,19].

It should be observed that, although a careful construction of the overlay helps increasing the efficiency of P2P systems for wireless ad hoc networks, the combination of node mobility, lack of infrastructure and unreliable wireless links renders the current proposals unsuitable for application in large ad hoc networks. Typically, it is assumed that the P2P network is formed by a few tenths of nodes (see [6,10,14]). Hence, the design of a *scalable* P2P system for wireless ad hoc networks is still an open problem.

The DHT algorithm proposed in this paper is a variation of the Viceroy algorithm proposed in [12] based on the use of geographic information. For this reason, we call our algorithm *Georoy*. Another major difference between Viceroy and Georoy is that the former implements a flat peer-to-peer network (i.e., peer nodes both provide content – resources – to the network and implement a distributed catalog for indexing), while the latter is based on a two-tier architecture: lower tier nodes (leaf peers) provide content, and upper tier nodes (super peers) implement a distributed catalog of available resources. In our context, leaf peers are mobile users, which make they resources available to the community, and super peers are the wireless routers (also called *wireless community nodes* in the following) composing the infrastructure of the wireless community mesh. The use of a DHT at the super peer level allows us to implement a very efficient indexing of the available resources: every super peer maintains virtual links only to a constant number of neighbors (at most 7, independently of the system size), and queries about resource localization can be correctly answered by traversing at most $O(\log n)$ links in the virtual overlay at the super peer level, where n is the number of super peers in the network. Furthermore, our design inherits from Viceroy important properties such as load balancing in the overlay network, and efficient maintenance of the distributed indexing mechanism in presence of super peer and leaf peer join/leaves.

Another major contribution of this paper is a theoretical analysis of the *stretch factor* of the overlay network built by Georoy. Informally speaking, the stretch factor measures how close a virtual overlay is to the topology of the underlying network (see Section C.3 for a formal definition): the lower the stretch factor, the closer the virtual overlay to the physical underlying network. Under the assumption that super peers are distributed uniformly at random in a square region, we formally prove that the stretch factor of our system is at most $O(\sqrt{n \log n})$, showing that the virtual overlay closely matches the underlying network topology. Up to straightforward modifications, our bound on the stretch factor can be applied to any DHT approach which maps keys in the [0,1] interval, such as Chord [21] and Koorde [9]. We observe that proving similar properties in traditional, Internet-based P2P systems is very difficult, since little is known about the actual topology of the Internet. Also, no similar bound has been so far proved for wireless ad hoc or sensor networks.

C The proposed Location-aware DHT algorithm

The algorithm used to implement the DHT abstraction at the super peer level is an adaptation of the Viceroy algorithm introduced in [12] to the wireless mesh network scenario. For this reason, we describe Viceroy before introducing our algorithm.

C.1 The Viceroy algorithm

In Viceroy both peer IDs and resource keys are mapped by a hashing function into to the same ID space (metric), i.e. the unit ring $[0, 1]$. Each key resides on the peer with the smallest ID larger than the key ID, i.e. the key range associated with peer p comprises all the resource keys with ID smaller than p and larger than the ID of the predecessor of p in the unit ring (see Figure F.1). For simplicity, from now on we use notation p to denote both a generic peer and its ID in $[0, 1]$, and notation k to denote both a generic resource key k and its ID in $[0, 1]$. Note that two keys that are close in the ID space are located on peers which are also close in the same metric space.

Viceroy's overlay network uses both "short" and "long" range links between peers, which are established by combining the unit ring topology with an approximation of the butterfly network[3]. In order to emulate the butterfly network, each peer is assigned a certain level in the network, i.e. the identity of a node in the network is composed of the pair (p, l_p), where p is the peer ID and l_p is the level of node p in the butterfly.

C.1.0.1 ID and level assignment. When joining the network for the first time, a SP p chooses uniformly, at random, a value in the $[0, 1]$ interval, which represents its ID. Then, p randomly chooses its level l_p in the butterfly. Ideally, a peer should select its level by choosing uniformly at random a number in $\{1, \ldots, \log n\}$, where n is the number of nodes currently forming the network. Since exactly computing n is virtually impossible in practice, the following procedure is used to compute an approximation of n. When joining the network, peer p first computes the distance $d(p, succ(p))$ to its successor in the

[3] The butterfly network on n nodes is a multi stage network with $\log n$ stages, where a node at stage i is connected with a limited number of nodes at stages $i - 1$ and $i + 1$. For a description of the butterfly network, see [20].

unit ring by invoking a LOOKUP(p,p) operation (see below for a description of the LOOKUP procedure); then, it estimates n as $n_0 = 1/d(p, succ(p))$, and selects a level by picking uniformly at random a number in $\{1, \ldots, \log n_0\}$.

While the peer ID does not change during network lifetime, the peer level can change in order to maintain a balanced subdivision of nodes into stages when new peers join/leave the network. More in particular, a peer must recompute its level when its successor in the unit ring (and, consequently, its estimation of n) changes.

C.1.0.2 Overlay construction. Viceroy's overlay network uses three types of directed links: *i*) *unit-ring links*, which connect a peer with its predecessor and its successor in the unit ring; (*ii*) *level-ring links*, which are used to form a virtual bi-directional ring between the peers at the same level; and *butterfly links*, which are used to emulate a butterfly network. Butterfly links are composed of an *upward* and two *downward* links. The upward link connects peer p at level $h > 1$ to the first $(h-1)$-level peer after position p on the unit ring. The downward left link (the *short range* link) connects p to the first $(h+1)$-level peer after position p on the unit ring; the downward right link (the *long range* link) connects p to the first $(h+1)$-level peer after position $p + 1/2^h$ on the unit ring. Summarizing, every peer in the Viceroy overlay network has at most 7 outgoing links: 2 unit-ring links, 2 level-ring links, and 3 butterfly links (see Figure F.2).

C.1.0.3 Routing. Routing in Viceroy is essentially performed by invoking a LOOKUP(x, y) function, where x is the requested key or peer ID (we recall that both resource keys and peer IDs are mapped in the same metric space), and y is the ID of the peer that invoked the function. The result of a LOOKUP(x, y) operation is the value associated with key x, or the ID of the peer that manages the key range to which x belongs (i.e. the peer with smaller ID larger than x) if no value is associated with x. When the peer y at level l_y needs to retrieve key x, it initializes the current position to y and invokes the LOOKUP(x, y) function. The LOOKUP request is routed in the overlay, using the following three-phased process:

(1) *up to the root*: starting from y, the request is recursively forwarded upward in the butterfly - each time updating the current position - using the upward link, until level 1 is reached;
(2) *traverse the tree*: the request is forwarded downward in the butterfly, using either the short or the long range link depending on whether x is at distance smaller than $1/2^h$ from the current position or not;
(3) *traverse the ring*: finally, when the current peer has no downward links or it overshoots the target x, the request is forwarded using the level-ring

and/or unit-ring links, until the peer s that manages the key range to which x belongs is found. Node s then returns the answer to the LOOKUP operation to y.

C.1.0.4 Overlay maintenance. A peer y in the overlay maintains the following information: (1) the ID on the unit ring (for simplicity, y); (2) the current level l_y; (3) the connections on the unit ring, $pred_y$ and $succ_y$; (4) the connections on the level ring, $pred_y^l$ and $succ_y^l$; (5) the upward butterfly connection, $succ_y^{l-1}$; and (6) the downward butterfly connections, $short_y^{l+1}$ and $long_y^{l+1}$.

When a new peer y joins the network, it first selects its ID as described above. By invoking LOOKUP(y, y) [4], node y finds its successor $succ_y$ in the ring, and establishes a connection to it. By exchanging information with $succ_y$, peer y knows the ID of its predecessor $pred_y$ in the ring, and establishes a connection to it. Both $succ_y$ and $pred_y$ update their predecessor and successor link, respectively, in order to reconfigure the correct links in the unit ring. Then, $succ_y$ transfers to y all the key-value pairs whose key is between $pred_y$ and y. This completes the unit ring update. After that, peer y selects the current level l_y in the butterfly as described above. Then, it finds its successor $succ_y^l$ and predecessor $pred_y^l$ in the level ring by single-stepping on the unit ring, and establishes connections with them. The predecessor and successor links in the level ring of peers $succ_y^l$ and $pred_y^l$ are updated accordingly. Finally, node y establishes the butterfly links by finding $succ_y^{l-1}$ and $short_y^{l+1}$ (this can be done by single stepping on the unit ring), and $long_y^{l+1}$ (this can be done by invoking LOOKUP$(y + 1/2^{l_y}, y)$, and then single stepping on the unit ring).

When peer y leaves the network, it has to remove all its established connections, notifying all neighbors in the overlay to update their links; then, y transfers its content to its successor in the unit ring.

When the current level changes (we recall that this is possible if $succ_y$ changes), peer y has to update its level ring connections and butterfly connections, notifying the neighbors when necessary.

C.1.0.5 Viceroy's properties. The following properties of Viceroy have been proved in [12], under the assumption that peer IDs and resource keys are distributed independently and uniformly at random in $[0, 1]$:

[4] Given the assumption of uniform ID distribution, the probability of two peers having the same ID is close to 0.

– *dilation*: If n peers are present in the network, then a LOOKUP operation is successfully completed by traversing at most $O(\log n)$ links w.h.p.[5]
– *congestion*: Let the *load* of a peer be the probability that it is involved in a LOOKUP operation on a random value generated at a random starting point, and let the *congestion* of the network be the maximum of the peer loads. If n nodes are present in the network, the expected load for any peer is $O(\log n/n)$, and the congestion is $O((\log^2 n)/n)$ w.h.p.
– *node degree*: If n peers are present in the network, then the out-degree of each node is at most 7, the expected in-degree is $O(1)$, and the largest in-degree of a peer is $O(\log n)$ w.h.p.

C.2 The Georoy algorithm

In this sub-section we show how to adapt Viceroy to a wireless mesh network scenario. First, we observe that, differently from Viceroy, in Georoy we assume a hierarchy of peers. Network members, which represent the lower tier of the hierarchy, are denoted as leaf peers (LPs) and provide content; the higher tier of the hierarchy is, instead, composed of super peers (SPs) which implement a distributed index of the available resources. Hence, in Georoy we assume that identities (i.e., a peer ID in the [0,1] interval, and a level in the butterfly) are assigned only to super peers. Also, a resource ID is assigned to each resource made available in the network; the resource ID is mapped into [0,1] by an appropriate hash function. In Georoy, the answer to a lookup operation on a certain resource k contains the location of the resource (e.g., the addresses of the leaf peer which holds k and its Home SP) in case k is available. We distinguish between two types of unavailability of the requested resource: *temporary unavailability* (i.e., the resource is available at some LP, u, in the network, but u is currently disconnected from the network), and *permanent unavailability* (i.e., none of the LPs has the requested resource). The answer to the lookup operation on k could be the address of $p^{(H)}(u)$ in case of temporary unavailability, and the ID of the super peer that manages the key range to which the key associated with k belongs in case of permanent unavailability.

In what follows, we assume that peers[6] are distributed in a square deployment region of side s, for some constant $s > 0$, i.e. peers are located in[7] $R = [0, s)^2$. Furthermore, we assume that peers (i.e., wireless community nodes) are aware of their position in R.

[5] W.h.p. means with probability at least $1 - c/n$, for some constant $c > 0$.
[6] From now on in this Section, the term ‘peer’ is used instead of the term ‘super peer’.
[7] The use of a right open interval is needed to simplify the definition of the ID mapping function below, and it has no practical consequences.

Our goal is to define a mechanism to assign peer IDs that preserves geographical proximity, i.e. two peers which are geographically close should be assigned close IDs in the unit ring. Preserving proximity is fundamental to achieve a close correspondence between the virtual overlay and the physical network topology.

In the Georoy algorithm, peer IDs are computed as follows. Let (x, y) denote the coordinates of peer p in R. We define a mapping function $\mathcal{M}$ that maps a point in R into $[0, 1]$ as follows:

$$\mathcal{M}(x,y) = \begin{cases} \frac{x\Delta}{s^2} + \lfloor \frac{y}{\Delta} \rfloor \cdot \frac{\Delta}{s} & \text{if} \lfloor \frac{y}{\Delta} \rfloor \text{ is even} \\ \\ \frac{(s-x)\Delta}{s^2} + \lfloor \frac{y}{\Delta} \rfloor \cdot \frac{\Delta}{s} & \text{if} \lfloor \frac{y}{\Delta} \rfloor \text{ is odd} \end{cases}, \qquad \text{(C.1)}$$

where Δ is an arbitrary constant with $0 < \Delta < s$.

The intuition behind our mapping function is depicted in Figure F.3: the deployment region R is divided into s/Δ sub-regions of equal area, which are defined in terms of the y coordinate. All the nodes in the same sub-region are mapped into the same segment of the unit ring, where the position of the node within the segment is determined by its x coordinate. In order to preserve proximity, the order of nodes in a segment is reversed alternately (see Figure F.3).

Before ending this subsection, we observe that the above defined mapping can be applied to any DHT approach which maps resource and node IDs in the [0,1] interval, such as Chord [21] and Koorde [9].

C.3 Georoy analysis

We first show that Georoy preserves all the nice properties of the Viceroy algorithm described in Section C.1, under the assumption that wireless community nodes are distributed independently and uniformly at random in $R = [0, s)^2$. This fact is a straightforward consequence of the following theorem, which shows that the function $\mathcal{M}$ defined in the previous section maps a two-dimensional uniform distribution of wireless community nodes (peers) in R into a uniform distribution of peer IDs in $[0, 1]$.

Theorem 1 *Assume the wireless community nodes (peers) are distributed independently and uniformly at random in $R = [0, s)^2$; then, the peer IDs computed according to mapping $\mathcal{M}$ are distributed independently and uniformly at random in $[0, 1]$.*

Proof 1 [SKETCH] *Let $S_{x,d} = [x, x+d]$, with $0 \leq x \leq 1-d$ and $0 < d \leq 1$, be an arbitrary segment of length d in $[0,1]$, and let $\mathcal{M}^{-1}(S_{x,d})$ be the set of points in R which are mapped to $S_{x,d}$ by using $\mathcal{M}$. It is easy to see that the area $|\mathcal{M}^{-1}(S_{x,d})|$ of the region $\mathcal{M}^{-1}(S_{x,d})$ depends only on d, i.e. segments of equal length d in $[0,1]$ correspond to regions of the same area $|\mathcal{M}^{-1}(S_{x,d})| = f(d)$. From this fact, and from the assumption that wireless community nodes are distributed independently and uniformly at random in R, it follows that peer IDs are distributed uniformly at random in $[0,1]$.*

Corollary 2 *Georoy preserves the properties of dilation, congestion, and node degree of Viceroy.*

We now prove an upper bound on the stretch factor of our algorithm, which is formally defined as follows:

Definition 3 (Stretch factor) *Given a query on key k, let $l(k)$ be the hop distance in the physical network between the peer at which the query is originated and the peer that manages the key range to which k belongs; furthermore, let $P(k)$ be the path traversed by the query on k in the overlay network, and let $l(P(k))$ be hop length of $P(k)$ in the physical network. The stretch factor is:*

$$stretch(k) = \frac{l(P(k))}{l(k)} \ . \tag{C.2}$$

In order to prove the bound, we first show that a 1-hop path in the overlay network corresponds to a path with at most $\sqrt{\frac{n}{\log n}}$ hops in the physical network.

Lemma 1 *Assume n wireless community nodes, each with transmitting range r, are distributed independently and uniformly at random in $R = [0,s)^2$. Furthermore, assume that $r = 2s\sqrt{\frac{2\log n}{n}}$. Then, a 1-hop path in the overlay network corresponds to a path with at most $\sqrt{\frac{n}{\log n}}$ hops in the physical network, a.a.s.*[8]

Proof 2 *Let us divide R into non-overlapping square cells of equal side $h = \frac{r}{2\sqrt{2}}$. The value of h is chosen so that, independently of its location, any node in a cell has a direct communication link to every other node in its cell and in the neighbor cells (horizontal, vertical, and diagonal adjacency). Thus, we have a total of $N = \frac{8s^2}{r^2}$ cells. Under the assumption that $r = 2s\sqrt{\frac{2\log n}{n}}$, we can use occupancy theory (Th. 1, pg. 5 of [11]) to prove that every cell contains at least one node a.a.s. We observe that the longest possible 1-hop path in the overlay network has length $s\sqrt{2}$ (which corresponds to the diagonal of R), and*

[8] A.a.s. (Asymptotically Almost Surely) means with probability that converges to 1 as $n \to \infty$.

that this length is covered by traversing at most $\sqrt{N} = \sqrt{\frac{n}{\log n}}$ cells. Since every cell contains at least one node a.a.s., we have that the progress towards the destination at each hop in the physical network is of at least one cell a.a.s. This completes the proof of the lemma.

Note that Lemma 1 requires a condition on the transmitting range of the community nodes, which implies that the physical network must be relatively dense: every community node has $O(\log n)$ neighbors on the average. On the other hand, it is well know that this is the minimum possible density that is required to have an a.a.s. connected network under the assumption of uniformly distributed nodes (see, for instance, Th. 4.1.1, pg. 42 of [18]). Since in community networks having a connected network at the wireless router level is fundamental to provide community-related services, the condition of Lemma 1 is not stringent.

Theorem 4 *Under the assumptions of Lemma 1, the stretch factor of the Georoy algorithm is $O(\sqrt{n \log n})$ a.a.s.*

Proof 3 *By the dilation property of Georoy, we have that a query traverses $O(\log n)$ hops in the overlay network w.h.p. By Lemma 1, each hop corresponds to at most $\sqrt{\frac{n}{logn}}$ hops in the physical network a.a.s. Hence, a query traverses at most $O(\sqrt{n \log n})$ hops in the physical network a.a.s. The proof follows by observing that we have $l(k) \geq 1$ for any possible query on key k.*

D Mobility management procedures

In this section we describe the procedures required to integrate the Georoy DHT algorithm in a realistic system where any node can suddenly show up, move or disappear. We remark that the type of dynamic behavior addressed in this section is different from the one addressed in the original version of Viceroy. In [12], the authors present procedures for dealing with dynamic conditions at the overlay network level, which corresponds to the upper tier of the hierarchy in our design. In this section, we complement these procedures (which we have briefly discussed in Section C.1) with similar procedures necessary to deal with dynamic conditions at the lower tier of the architecture (leaf peers).

First, we note that at a certain time instant, t, the generic LP u may periodically pass from *active state* to inactive and viceversa. Accordingly, a joining and a leaving procedure are required. This is described in Section D.1. The procedures required to update the distributed resource catalog when a LP decides to share a new resource, or not to share a certain resource anymore, are presented in Section D.2. Section D.3 describes how the retrieval of resources is addressed. Finally, the handoff of LPs from the formerly responsible SP to another is dealt with in Section D.4.

D.1 LP joining/leaving procedures

When a new LP u passes from the inactive to the active state, a *joining operation* is needed, i.e., node u must connect to one SP in its proximity and the information about the resources available in the distributed catalog must be updated.

To this end, LP u first listens in the wireless interface if there is any SP in its radio coverage. If at least one SP is heard, then u selects the one with the highest signal-to-noise ratio, which we denote as $p(u)$, and registers to it. Upon registering, LP u provides $p(u)$, which is called the *responsible SP* of node u, with the list of the resources it is willing to share. Such information is maintained up to date by $p(u)$ in a local database of avalaible resources. Accordingly, any SP, p, has a list of the resources shared by all the LPs it is responsible for, U_p.

Node u stores another list with the resources it was willing to share the last time it was connected to the network. If there are changes, i.e., leaf peer u wants to share new resources and/or does not want to share certain resources which are in the above list, then it must inform the Home SP $p^{(H)}(u)$ according to the procedure described in Section D.2.

When a LP u leaves the community network, the list of resources available in the network has to be updated. To this purpose, node u notifies its responsible SP, $p(u)$, before leaving the network, and $p(u)$ puts the resources shared by u in *park mode* through an appropriate tagging of the entry in its local database. Also the Home SP, $p^{(H)}(u)$, must be informed that u is leaving the network so that it puts in the *park mode* the resources shared by u.

As a consequence, if u joins again the same SP, the only operation required is to move the reources shared by u in the *available mode* through a de-tagging of the relevant entries of the local resource database in $p(u)$ and in the catalog stored at $p^{(H)}(u)$.

In this way the signaling in the network is maintained at a minimum level. Resources that are in *park mode* for a time interval longer than a given threshold are removed from the local resource database, and considered as no longer available. The IDs of these resources are notified by $p^{(H)}(u)$ to the SPs that manage the corresponding key ranges, so that they can decide whether to remove the corresponding resource ID from the distributed catalog (according to whether the same resource has re-appeared or not in another part of the network).

Note that a LP u can also detach from the network without notifying its responsible SP. Such event can be detected by the network as follows. If for a certain time interval, τ_D, the SP $p(u)$ does not receive any query from u, then it sends a beacon to u. If $p(u)$ does not receive any answer from u within a certain time, it labels the resources shared by LP u as in *park mode* and informs $p^{(H)}(u)$.

D.2 Insertion/removal of resources to/from the distributed catalog

Suppose that LP u wants to share a new resource in the community. There are two cases:

a $p(u) = p^{(H)}(u)$. In this case, node u informs node $p(u)$. This node evaluates the key k identifying the new resource and forwards all information required to localize k to the SP which is responsible of managing the corresponding key range.
b $p(u) \neq p^{(H)}(u)$. In this case, node u informs node $p(u)$, which inserts the new resource in the catalog of the locally available resources and then informs $p^{(H)}(u)$. The latter next updates the distributed catalog as in the previous case.

Observe that in case b) the utilization of the Home SP requires a further step, i.e., to inform node $p^{(H)}(u)$. Although this produces some overhead, the Home SP mechanism achieves better performance in case of LP mobility, as will be described in Section D.4 and confirmed by the simulation results shown in Section E.

Similar procedures are executed when a node is not anymore willing to share a certain resource.

D.3 Information retrieval

Search requests are issued at the lower tier, and are routed in the overlay at the upper tier. When a LP u issues a request for a certain resource, it forwards such request to its SP $p(u)$. The SP $p(u)$ first checks whether the request can be satisfied locally using the local available resources database[9]. If the request cannot be satisfied locally, i.e., the corresponding resource is not stored by any

[9] In a follow up paper, we will address the problem of having multiple copies of the same resource in the network labelled with different keys. Moreover, we will also deal with the problem of multiple copies of the same resource characterized by

of the LP belonging to $U_{p(u)}$, then node $p(u)$ initiates a search in the overlay network as described in Section C.2. Let v denote the LP storing the requested resource. The result of the algorithm is the identifier of the Home SP of v, that is $p^{(H)}(v)$.

The request will thus be forwarded towards $p^{(H)}(v)$, which stores information about the location of v. If the resource is available the requesting node u is informed about the identity and position of v, i.e., its address and responsible SP. If the resource is currently in *park mode*, the SP $p^{(H)}(v)$ informs node u that the requested resource is currently not available and u can decide whether to wait until the resource becomes available or not. In the former case, SP $p^{(H)}(v)$ informs u when the resource is available again and the resource transfer can begin[10].

If the LP v moves or switches off during the resource transfer, the download stops and a new resource request procedure has to be initiated. When the resource is found again, download can be completed. In this context, fragmentation mechanisms could be used so that download of only the missing fragments is required, thus increasing efficiency.

Finally, if the requesting LP, u, moves or switches off during the resource transfer, the download stops and is restored when u becomes available again.

D.4 LP handoff management

Suppose that a certain LP u, which was formerly associated with SP p', migrates in the coverage area of another SP, p''. In this case the following operations are required: (i) informing node $p^{(H)}(u)$ that from now on $p(u) = p''$; (ii) deleting the resources stored by u from the catalog of the resources locally available at p'; (iii) inserting the resources stored by u into the catalog of the resources locally available at p''; and (iv) informing all the LPs that are currently downloading resources from u, if any, that this node has moved to another position.

Observe that the use of the Home SP mechanism increases efficiency significantly when handoff occurs. In fact, besides local signaling between the leaf peer u and the past and current responsible SPs, p' and p'', only a location update must be sent to the Home SP, $p^{(H)}(u)$. Instead, if the Home SP mechanism was not used, the location update should have to be trasferred to all SPs that contain the location information concerning node u.

different reliability levels so that a priority choice can be done.

[10] This is typically done in peer-to-peer applications.

E Performance evaluation

We consider a wireless mesh community network organized into two-tiers consisting of N_{SP} super peer nodes and N_{LP} leaf peer nodes, with $N_{SP} \in [16, 25, 36, 49, 64, 81, 100, 121, 144]$, and $N_{LP} = 1000$, respectively.

The network area is supposed to be a square 1000x1000 m^2 large. Furthermore, LP nodes are assumed to be distributed uniformly at random, while SP nodes are assumed to be distributed either regularly over a grid, or uniformly at random. In the first case the coordinates of the i-th SP, (x_i, y_i), are given by:

$$x_i = \begin{cases} \left(i - \lfloor\frac{i-1}{\sqrt{N_{SP}}}\rfloor \cdot \sqrt{N_{SP}} - \frac{1}{2}\right) \cdot \Delta & \forall i \in [1, N_{SP}] : \lfloor\frac{i-1}{\sqrt{N_{SP}}}\rfloor \text{ is even.} \\ s - \left(i - \lfloor\frac{i-1}{\sqrt{N_{SP}}}\rfloor \cdot \sqrt{N_{SP}} - \frac{1}{2}\right) \cdot \Delta & \forall i \in [1, N_{SP}] : \lfloor\frac{i-1}{\sqrt{N_{SP}}}\rfloor \text{ is odd.} \end{cases} \tag{E.1}$$

and

$$y_i = \lfloor\frac{i-1}{\sqrt{N_{SP}}}\rfloor \cdot \Delta + \frac{\Delta}{2} \tag{E.2}$$

where $s = 1000$ m and we set $\Delta = s/\sqrt{N_{SP}}$. As a consequence, applying eq. (C.1), we obtain that the identifier of the n-th SP is given by:

$$p_n = \mathcal{M}(x_n, y_n) = \left(n - \frac{1}{2}\right) \cdot \frac{1}{N_{SP}} \tag{E.3}$$

Furthermore, we assume that SPs p_i and p_j, with $i, j \in [1, N_{SP}]$ and $i \neq j$, located in the positions (x_i, y_i) and (x_j, y_j), respectively, are connected by a link if and only if one of the four following conditions hold: (a) $x_i = x_j$ and $y_i = y_j + \Delta$; (b) $x_i = x_j$ and $y_i = y_j - \Delta$; (c) $x_i = x_j + \Delta$ and $y_i = y_j$; or (d) $x_i = x_j - \Delta$ and $y_i = y_j$.

In this section we will focus on the performance of Georoy and compare it to Viceroy. Here the performance metrics under investigation will be the number of logical links in $P(k)$, i.e., the path traversed by a query on a generic key k in the overlay network, the number of hops in the physical network of $P(k)$, i.e., $l(P(k))$, and the stretch factor defined as in (C.2). We remark that we did not compare the information retrieval efficiency (i.e., number of successfully answered queries over total number of queries) of the two approaches since, under this respect, Viceroy and Georoy share the same design and, hence, performance. In fact, Georoy uses the same routing scheme as the original Viceroy algorithm, and it is the routing scheme that is responsible of information retrieval efficiency. On the other hand, as the simulation results reported in this section outline, Georoy and Viceroy show very different performance in terms of network load generated by queries.

Observe that both Georoy and Viceroy only consider SPs, therefore a variation in the number of LPs does not affect the performance of the algorithm. This is why the number of LPs has been assumed to be constant and equal to 1000.

In Figure F.4, we compare the average number of logical links traversed by a query using Georoy and Viceroy algorithms vs. the number of SPs. Each result was obtained by averaging over 100 simulations. As expected, Georoy and Viceroy algorithms exhibit the same behavior in terms of logical links being traversed at the overlay network level by a query on a certain resource key. This is because the only difference between Viceroy and Georoy lies in the way the overlay network is mapped into the physical network and thus there are no differences at the overlay network level. The same consideration also applies to the variance of the number of logical links traversed by a query on a certain key, as shown in Figure F.5.

In Figure F.6 we show the average number of hops traversed in the physical network by a query on a certain key when Georoy and Viceroy algorithms are utilized. It can be observed that the Georoy algorithm drastically reduces the number of physical hops needed to localize the searched resource with respect to the Viceroy algorithm. This was an expected result, and is due to the mapping function used in Georoy to reduce the stretch factor.

To additionally highlight the effectiveness of Georoy when compared to Viceroy, in Figure **??** we show a comparison between the stretch factor of the two algorithms. As expected, the Viceroy algorithm has a stretch factor which can be as much as 7 times higher than that of Georoy. Once again, the reason of this has to be searched in the similarity between the overlay and the physical network obtained using the Georoy approach. In Figure **??** we also show the upper bound on the Georoy stretch factor as given in Theorem 2. Notice that the stretch factor of Viceroy is far above this bound. As a final remark, we observe that the distribution of the super peers storing the resource locations only marginally affects the performance of the system, as shown by the simulation results reported in Figures F.4-7.

F Conclusions and future work

In this paper, we have studied the properties of Georoy, a location-aware enhancement to the Viceroy algorithm, and verified through simulation that it achieves the goal of enabling efficient and scalable peer-to-peer resource sharing in wireless mesh networks. More specifically, we observed that Georoy efficiently exploits the underlying physical wireless network and achieves a search efficiency that is as much as 7 times better than that achieved by Viceroy.

There are several ways to extend the work presented in this paper, which we are currently investigating. For instance, we are considering resource replication techniques to improve resource availability, and studying what is the optimal strategy for locating resource replicas in the system. Also, we are considering strategies for cross-layer optimization of the Georoy algorithm which exploit the broadcast nature of the wireless communications.

References

[1] Gnutella protocol specification v0.4: `http://rfc-gnutella.sourceforge.net/developer/stable/index.html`.

[2] Gnutella protocol specification v0.6: `http://rfc-gnutella.sourceforge.net/developer/testing/index.html`.

[3] KaZaA Website: `http://www.kazaa.com`.

[4] I. F. Akyildiz, X. Wang, W. Wang, "Wireless Mesh Networks: A Survey", *Computer Networks Journal (Elsevier)*, Vol. 47, No. 4, pp. 445–487, Mar. 2005.

[5] R. Bruno, M. Conti, E. Gregori, "Mesh Networks: Commodity Multihop Ad Hoc Networks", *IEEE Communications Magazine*, Vol. 43, No. 3, pp. 123–131, Mar. 2005.

[6] M. Conti, E. Gregori, G. Turi, "A Cross-Layer Optimization of Gnutella for Mobile Ad Hoc Networks", *Proc. ACM MobiHoc*, May 2005.

[7] M. Denny, M. Franklin, P. Castro, A. Purakayastha, "Mobiscope: A Scalable Spatial Discovery Service for Mobile Network Resources", *Proc. International Conference on Mobile Data Management (MDM)*, 2003.

[8] K. Hildrum, J.D. Kubiatowicz, S. Rao, B.Y. Zhao, "Distributed Object Location in a Dynamic Networks, *Proc. ACM SPAA*, Aug. 2002.

[9] M.F. Kaashoek, D.R. Krager, "Koorde: A Simple Degree-Optimal Distributed Hash Table", *Proc. International Workshop on Peer-to-Peer Systems (IPTPS)*, 2003.

[10] A. Klemm, C. Lindemann, O.P. Waldhorst, "A Special-Purpose Peer-to-Peer File Sharing System for Mobile Ad Hoc Networks", *Proc. IEEE VTC-Fall*, Oct. 2003.

[11] V.F. Kolchin, B.A. Sevast'yanov, V.P. Chistyakov, *Random Allocations*, V.H. Winston and Sons, Washington D.C., 1978.

[12] D. Malkhi, M. Naor, D. Ratajczak, "Viceroy: A Scalable and Dynamic Emulation of the Butterfly", *Proc. ACM Symposium on Principles of Distributed Computing (PODC)*, Jul. 2002.

[13] Napster Website: `http://www.napster.com`.

[14] H. Pucha, S.M. Das, Y.C. Hu, "Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile Ad Hoc Networks", *Proc. IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, 2004.

[15] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker, "A Scalable Content-Addressable Network", *Proc. ACM Sigcomm*, Aug. 2001.

[16] S. Ratnasamy, M. Handley, R. Karp, S. Shenker, "Topologically-Aware Overlay Construction and Server Selection", *Proc. IEEE Infocom*, Jun. 2002.

[17] F. Sailhan, V. Issarny, "Scalable Service Discovery for MANET", *Proc. IEEE PerCom*, 2005.

[18] P. Santi, *Topology Control in Wireless Ad Hoc and Sensor Networks*, John Wiley & Sons, Chichester, UK, 2005.

[19] K. Seada, A. Helmy, "Rendezvous Regions: A Scalable Architecture for Resource Discovery and Service Location in Large-Scale Mobile Networks", *Proc. IEEE Workshop on Algorithms for Wireless, Mobile Ad Hoc and Sensor Networks (WMAN)*, 2004.

[20] H.J. Siegel, "Interconnection Networks for SIMD Machines", *IEEE Computer*, Vol. 12, No. 6, pp. 57–65, 1979.

[21] I. Stoica, R. Morris, D. Karger, M. Kaashoek, H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications", *Proc. ACM Sigcomm*, Aug. 2001.

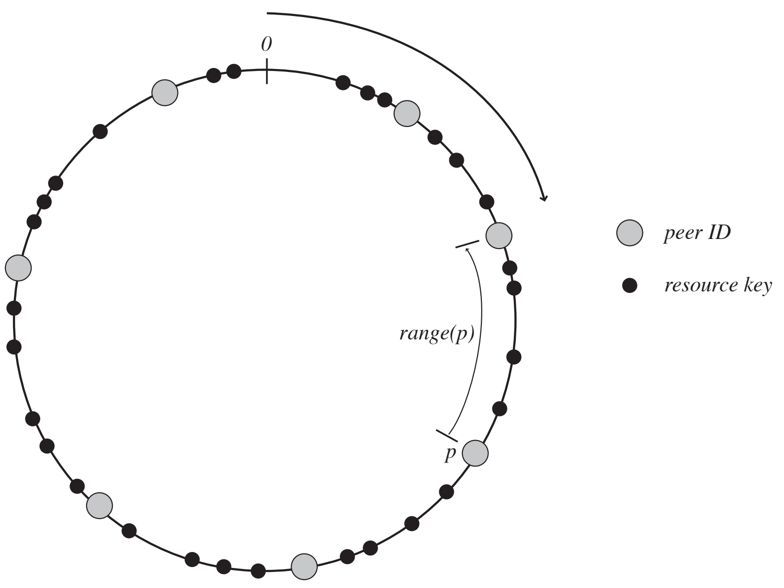

Fig. F.1. Viceroy's mapping of peer IDs and resource keys in the unit ring. Peer p manages all the resource keys in $range(p)$.

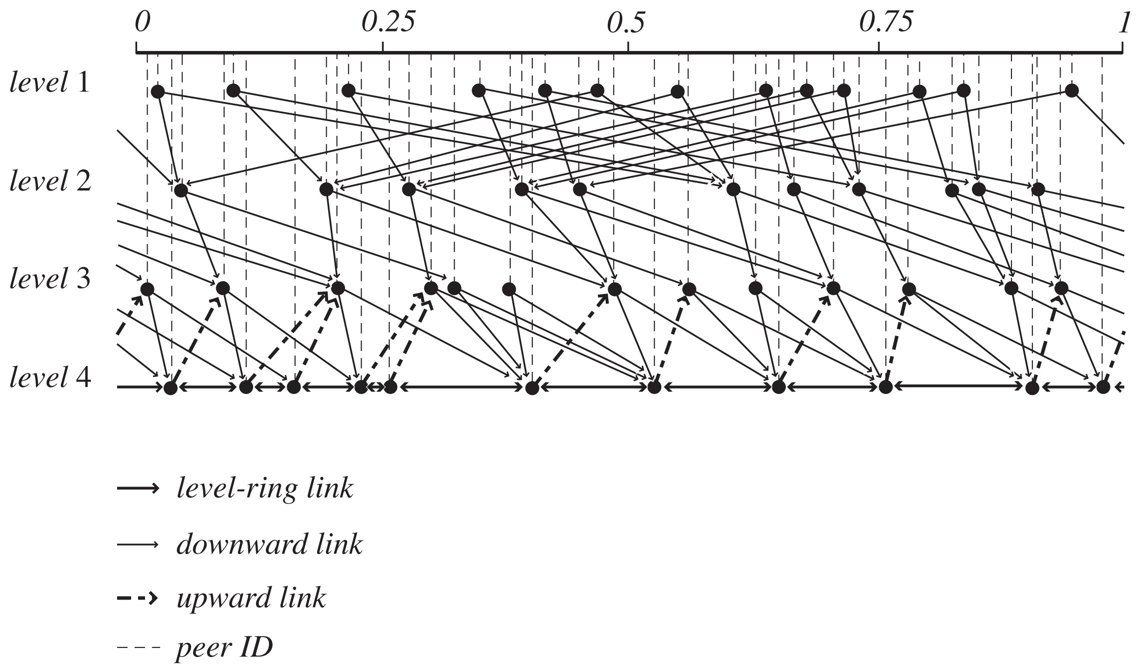

Fig. F.2. Viceroy's overlay network. For clarity, unit-ring links are not shown, and level-ring and upward links are shown only at level 4.

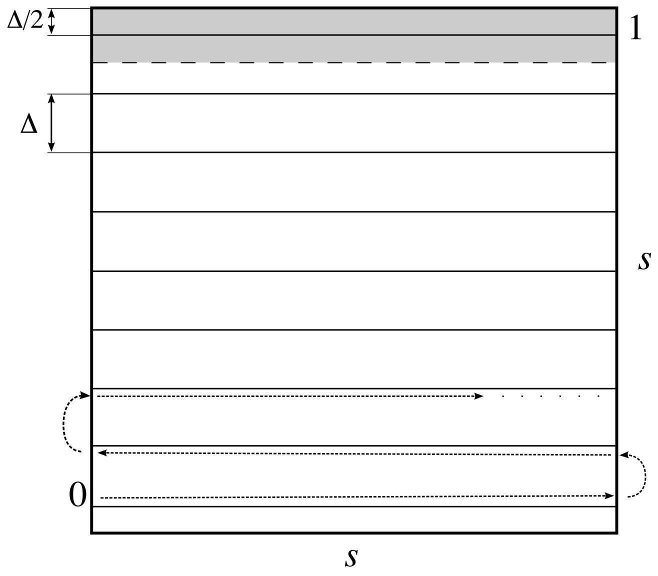

Fig. F.3. The mapping $\mathcal{M} : R$ is divided into sub-regions (shaded area), and nodes in the same sub-region are mapped into the same segment of the unit ring. The order of nodes in a segment is reversed alternately to preserve proximity.

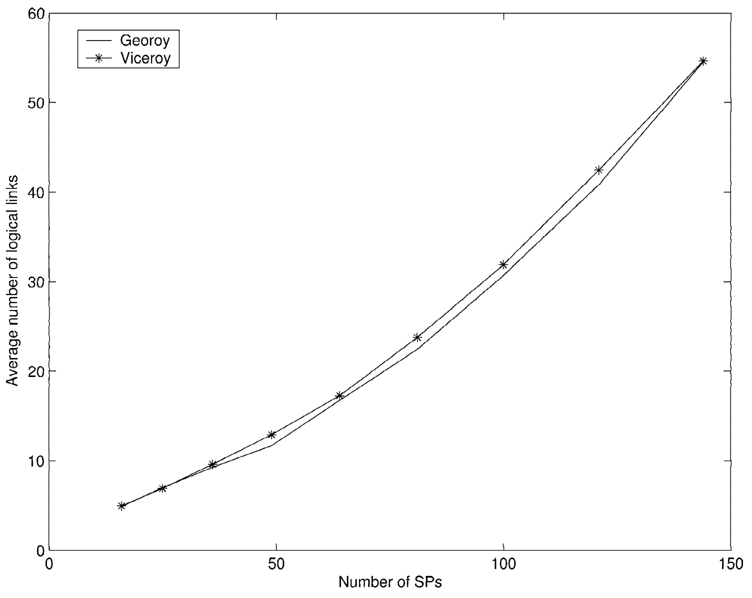

Fig. F.4. Comparison between the average numbers of logical links in the overlay network level in Georoy and Viceroy.

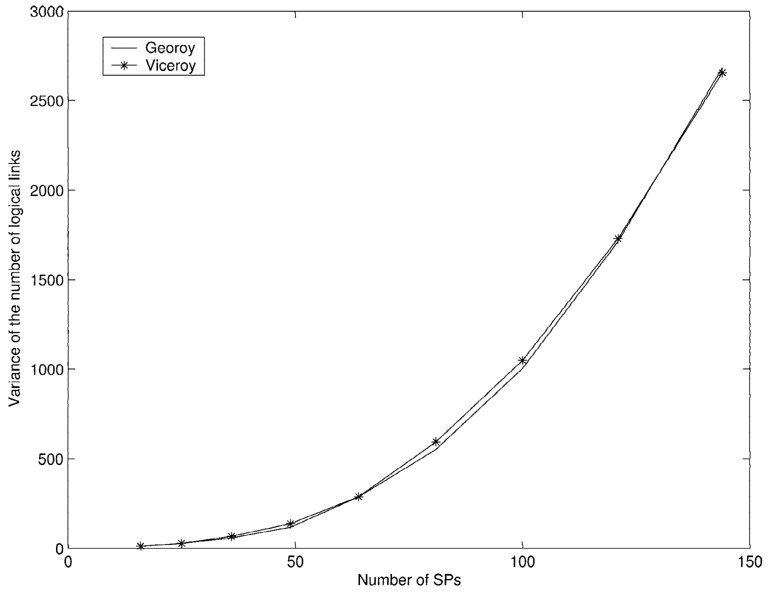

Fig. F.5. Comparison between the variance of the numbers of logical links at the overlay network level in Georoy and Viceroy.

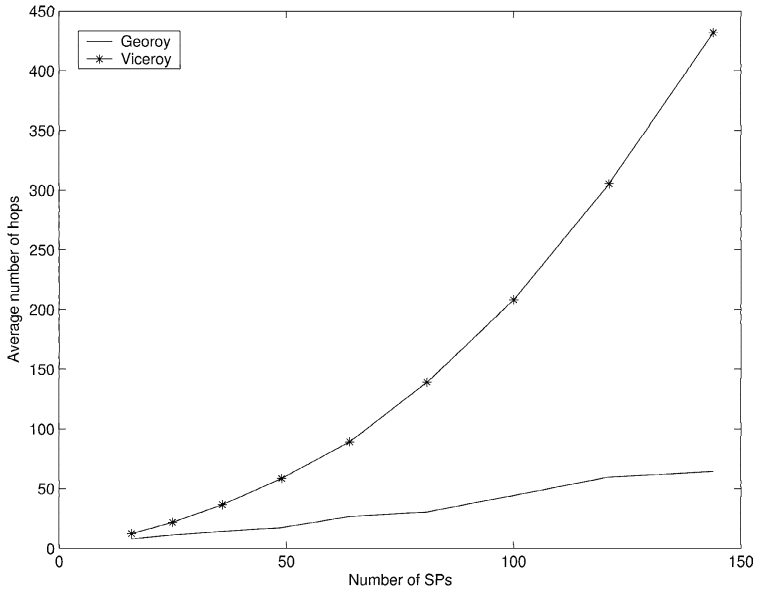

Fig. F.6. Comparison between the average number of hops in the physical network in case of Georoy and Viceroy.