Performance Evaluation of Overlay-Based Range Queries in Mobile Systems

Amine M. Houyou, Alexander Stenzer, and Hermann De Meer

University of Passau, Faculty for Mathematics and Computer Science,
Chair for Computer Networks and Communication, Innstr. 43
94032 Passau, Germany
{houyou, stenzer, demeer}@fim.uni-passau.de

Abstract. Current mobility management systems are operator centralized, and focused on single link technologies. In heterogeneous wireless mesh networks, vertical handovers could be a lengthy procedure. In order to support context-aware handovers between heterogeneous wireless cells, the mobile user needs to access information managed by administratively separate domains. Handover does not occur based on blind discovery mechanisms on the link layer but based on a pre-discovery of the wireless mesh topology using a context-aware search process. The mobile user has access to an overlay which forms a middleware that connects separate management domains, while allowing the user to retrieve uniform descriptions of underlying access technologies. Based on the topology information and access capabilities of the user, a context-aware handover is carried out. The overlay queries are sent to edge management servers to retrieve the cells' status as near to real-time as possible. The query system uses the geographic context to both address the managed objects as well as structuring the overlay. An analytic study and simulation are used to quantify the communication overhead of such a range query. A design methodology is also extracted from both studies.

1. Introduction

The wireless infrastructure so far has been laid out by networks operators in order to reach their customers through investing into well-organized and hierarchically structured cellular networks. The transition between wireless cells is closely monitored and tracked by the network. The location attribute of the mobile user is described through the cell's unique identifier and is in turn stored in a hierarchical structured data-base. Visited location registries (VLR) communicate with home location registries (HLR) to update the network as a whole of the location of each mobile node in the system. A mobile node only needs to report its current cell ID, for incoming calls to be directed to it. Some predictive measures could be also taken knowing the topology and direction of the movement of a mobile node 5. A home location registry (HLR) can also store the location updates (LUs), sent from a communicating terminal, or use paging to indicate to dormant devices their current cell IDs 2.

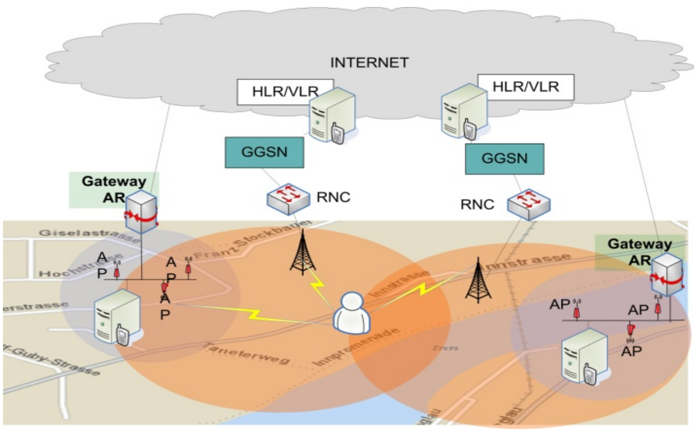

Fig. 1. The distributed nature of cellular and wireless networks management

Assuming that the mobile user can connect to any of the heterogeneous wireless mesh networks (WMN) 1 (example shown in Figure 1), the current handover process can only rely on link and network layer discovery mechanisms called beaconing. The cells and the mobile node beacon each other to discover movement. In a network centric approach, once the mobile node is located (i.e. the cell and the MN can see each other), a further assignment of resources could be done by the operator.

In the approach of this paper, the user can select out of several alternatives to which cell a handover should be started. Similar to the context-based handover advocated in 3, a mobile node could utilize the topology information to make a multiple criteria decision. This type of handover is triggered with the help of cross-layer mechanisms (see [[5]]).

Assuming the wireless mesh topology belongs to different domains, the problem of identifying this topology, from the mobile node's point of view, is that of querying each management domain separately for available cells. In other words, the mobile node requires a range query in a distributed and heterogeneous data base. The overlay can connect the data bases and mediate the data they manage through a single search process, which then transforms the problem to a single range query in an overlay.

The range query in overlay networks problem has two folds: first, the placement of the data items in the overlay network. Data items should be as close as possible to each other in the overlay as they are semantically. This minimizes the fragmentation of the search requests required to cover a given search or query box. The reader should be familiar with the basics of a DHT structure. DHTs split a continuous key space rather equally between participating peers, similar to the amount of routing information each peer stores locally. The hash values refer to an ID space (one-dimensional in Chord 14 for instance) called key space. It is worth noting that most structured P2P protocols have been optimized to solve an exact key match routing problem (called mediation). The continuous key space normally is matched to a random application level value (or objects), which allows a certain load balancing of both the distribution of application objects among the peers and the communication or mediation overhead.

If used unchanged, the mediation effort to launch a range query would first result into a splitting of the query to retrieve each single key contained in that range. Unless the data is structured in a way that two neighbouring keys on the P2P key space refer to two semantically close data items too, the resulting mediation overhead would quickly explode. The effort to refit the overlay structure to the data ranges results in a clustering of data among neighbouring peers. The overlay routing effort could be done once for each cluster. Reaching each cluster requires the largest lookup cost in terms of overlay hops. Recursively the remaining peers could be queried along the cluster. This type of research is not new and the reader could be further referred to 4 for a more generalized study.

This paper presents a design strategy for an overlay-based location-aware distributed data base problem. The overlay deals with the mediation problem, while the location-aware range query analysis shapes the structure of the overlay network. The paper is structured as follows. The related work is presented in Section 2. Section 3 explains the application domain and original architecture requirements for supporting context aware vertical handovers. The basic overlay requirements are explained too. Section 4 presents the asymptotic analysis of the range query cost. The design strategy is extracted from the analytic study. A simulative case study is presented in Section 5. Section 6 concludes the paper.

2. Related work

While mobility research has matured regarding cellular networks and even 3G and beyond scenarios 2, major efforts are still needed in WMNs and Ad-Hoc networks 1. Using overlays to manage mobility or as a support for mobility has also resulted in several proposals including earlier work by Stemm and Katz 12 supporting vertical handovers. I3-based protocol 14 followed as a large scale routing framework replacing mobile IP. Similar to the home and visited location registries the Palma P2P protocol manages the user's location 12. The work presented in this paper looks at supporting the mobile user itself in identifying the wireless diversity offered in 4G scenarios. The location management and mobility are not the focus here.

An important aspect of the proposed use of overlay is the support of location-based range queries. The mobile user, and based on its location, can query the location based service[1] 5 offered by the overlay network.

Range queries in P2P systems have addressed several issues such as non uniform partitioning of the object space. In this type of range queries, data ranges are neither continuous nor uniformly partitioned. This is the case for dictionary entries or names, where given ranges abruptly change. For this type of queries, several proposals have been made including Kademlia 7 and more systematically P-Grid based overlay 4. P-Grids are structured as prefix hash trees (PHT) (used in traditional database research). This suggests organizing the overlay as a data trie 4, which addresses nodes according

[1] Schiller defines Location-based services (LBS) as integrating geographic location with the general notion of a service 11

to a binary tree structure. For instance, a peer addressed with "001" would be on the route to all nodes or objects whose addresses share this prefix. In order to extend the scope of each peer at the lower part of the binary tree, these latter edge peers include pointer to other far off edge peers. Combined with search algorithms at the edge, the proposal in 4 is well suited for fragmented keyword based range queries. It is though worth noting that once at the edge the search algorithms are closer to flooding.

Another family of solutions is based on the multi-dimensional DHT CAN 9. In 9 a geographic coding is proposed. Although offering a better mapping between the multiple dimensions needed to describe a geographic zone, CAN based solutions partition the key space in multiple dimensional large zones, leading to flooding at the edge once reaching the zone. The dimension conversion is very restrictive, since once set, it is hard to encode anything else apart from zones, and therefore even for exact match query a range query is actually processed at the edge 4.

For this a Hilbert-based transformation function used in this paper has the advantage of offering both the possibility to convert any multi-dimensional attribute space to a one-dimensional hash value. It also allows an exact match query as well. Knoll et al. 6 have demonstrated the clustering property of the Hilbert Curve when applied to modelling geographic coding. Moon et al. 8 have made a pioneering study on the clustering property of the Hilbert-curve. And in this paper, the same space filling curve is used as a main cornerstone to design location-based distributed overlay systems.

3. Architecture

The proposed architecture could be used depending on the location awareness capabilities at the end-device. The terminal should be able to detect its capabilities and location, and to query the P2P architecture.

The main advantage of using DHTs in P2P applications is to provide a scalable system capable of storing large numbers of items while minimizing the routing cost and offering a load-balanced indexing of entries among peers. Chord, for instance, requires each peer node to store routing information for $O(logN)$ other peers, where N is the total number of participating peers. Similarly, it takes a maximum $O(logN)$ routing messages to find any content in a Chord ring.

A peer is a computational entity (which could be a virtual one) capable of mediating information about resources (in this case wireless cells and their status) stored or managed by the underlying server infrastructure (as shown in Figure 2). These peers are named access peers and their role is organizing the underlying infrastructure and clustering the data items managed by that infrastructure.

The peers which are accessible to the mobile users are called search peers. They can play the role of a proxy to the mobile user. They initiate the range query required for supporting the context-aware vertical handover. They also manage mobility in the sense that they redirect results of the query back to the user's current address. This could be similar to the I3 based host mobility solution, where the user is identified through a constant overlay ID and a variable value. This tuple is called a trigger and is stored transparently in the overlay network. Messages destined to the mobile node need just to reach the overlay node that stores the trigger. The latter overlay node is informed by the mobile node each time the IP address or location ID is changed. This indirection process, is transparent to the correspondent nodes, differently to Mobile IP, and is highly distributed (no central home agents anymore).

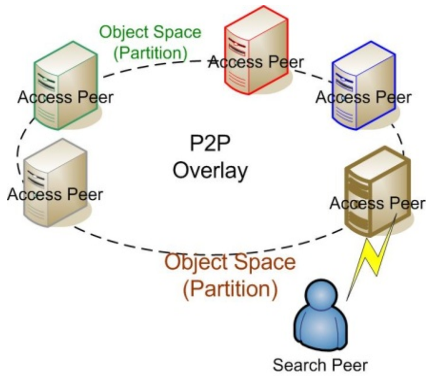

Fig. 2. P2P solution for organizing topology servers

Chord manages XML description files, retrievable by search peers and stored by access peers. The content or object space is no longer randomized through the use of a hash function such as *SHA-1*, distributing the keys among the peers in load-balanced way. In contrast, we rely for our system of Geocoding 11 to identify objects *(key,value)* through a geographic *key* pairs among peers. This could be compared to an online map service that can generate a zip code from entered GPS coordinates. The problem with Chord is that its query language requires a precise known object ID that can generate a precise (key, value) pair, making range queries difficult. For that reason, the ID is divided into blocks where the higher significant blocks represent unique larger areas of the earth surface, while the smaller significant blocks identify partitions of the large areas.

4. Analytic Study to Query Overhead

In order to achieve the sought distribution of data items among the peers, the following functions in a Chord like DHT need to be modified:

- Inserting an object in the overlay requires a coding function from the two-dimensional geographic coordinates to a single coordinate which preserves neighbourhood
- Query definition depends on a similar transformation function transforming a geographic range into the set of keys
- A set of neighbouring keys in the Chord space are easily converted back into a geographic range

For the above functionalities to be fulfilled, a space filling curve could be used.

4.1 Space Filling Curves

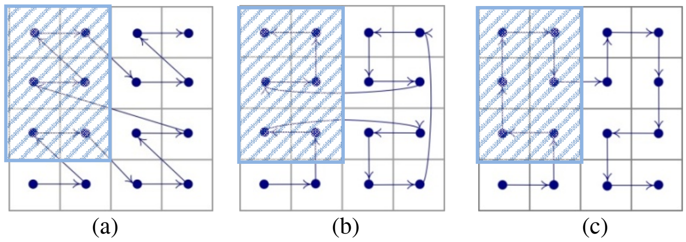

Fig. 3. 2-dimensional Peano space-filling curves: (a) z-curve, (b) Gray-coded curve, (c) Hilbert curve

Peano curves are space filling curves used in different domains from data compression to databases. Three Peano curve examples are illustrated in Figure 3. In the figure a 2-*dimensional* finite space has been divided in a grid of *4 × 4* square cells, and then filled by three different Peano curves. Each cell could be discretely addressed by x_n- and y_n-coordinates (e.g. the top left cells, covered by the query box, have the addresses *(0,1), (0,2), (0,3),(1,1),(1,2),(1,3))*. The Peano curve connects the grid cells with a directed line, which indicates the order of transition between the cells. Assuming that each cell is assigned an integer identifier, incrementally increasing following the shape and direction of the curve, the resulting new addresses are one-dimensional. Each 2-dimensional discrete coordinate pair (x_n,y_n) corresponds to a single integer ID. In general, a space filling curve is said to map a *d*-dimensional coordinate system (in the range $[0,2^{\gamma}-1]^{d}$) to a one-dimensional coordinate system ranging $[0,2^{\gamma\cdot d}-1]$, where γ is also known as the degree of the space filling curve. The degree of a curve relates to the size of the underlying *d*-dimensional grid. In a 2-dimonsional case, the filled space is a square surface divided in $2^{\gamma}\times 2^{\gamma}$ grid cells.

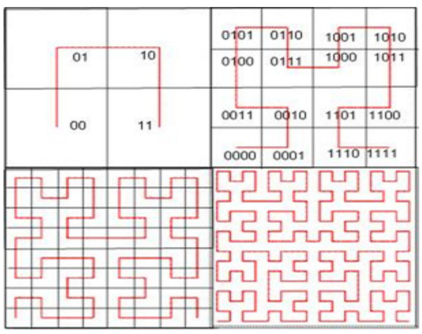

Fig. 4. γ = 1, 2, 3, 4 illustrations ($H_1^2, H_2^2, H_3^2, H_4^2$)

All Peano curves, including the Hilbert curve, have a fractal nature in the way they could be extended to describe the same space in a higher granularity. This depends on the degree of the space filling curve. In Figure 4, four examples are illustrated for degrees γ ranging from 1 to 4. The degree of the space filling curve affects the number of integers assigned to each cell. When represented in binary, this ID has to be $2 \cdot \gamma$-bits long.

Knoll et al. 6 have shown that out of the Peano space filling curves, the Hilbert curve maps geographic neighborhood best. In other words, given two neighboring binary addresses obtained following a Hilbert curve, these two points are geographically closest compared to other space filling curves. This property is also called clustering. "A Cluster is a group of grid points inside the query that are consecutively connected by a mapping (or a curve)" 8. In Figure 3, the Hilbert curve already shows that there is a single cluster covered by the query box (the 6 cells in the top left corner of each grid). On the other hand, the Gray-coded curve and Z-curve require two clusters for the same range box. It is shown in 8 that, in a 2-dimensional space, a range query is best described as a rectangular area. This self-organizing aspect allows any geographically close objects to be clustered together fulfilling our requirement for preserving the semantic neighborhood relation of the stored data items among our peers.

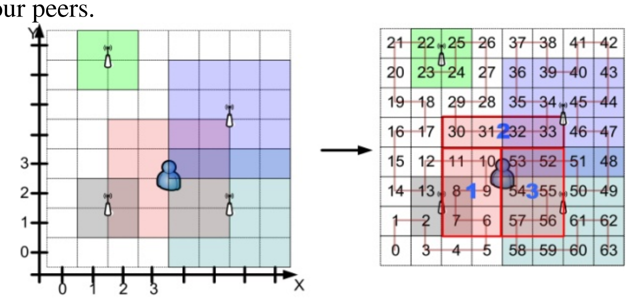

Fig. 5. Hilbert curve-based transformation from $(x,y)^{64}$ coordinates to 64 one-dimensional 6-Bit IDs ($d=2$ *and* $\gamma = 3$)

The example shown in Figure 5 illustrates the transformation of a 64-cell grid space addressed with (x_n, y_n) coordinates, where $(x_n, y_n) \in \mathrm{N} \mid 0 \leq x_n \leq 7, \, and \; 0 \leq y_n \leq 7$, As a result is $2^3 \times 2^3$ one-dimensional ID space requiring $3+3 = 6$ bits. In the shown example, there are *4* access cells requiring each a different number of keys to describe their geographic coverage. As an example "*Network 1*" (in Figure 5) requires four keys *22, 24, 25, 26*. The clustering effect is demonstrated by the given query box in the middle. To start the query, the start of the cluster and its length need to be identified. Here, three clusters exist *([6,11], [30, 33],* and *[52,57]),* requiring each a separate get(key) call. Identifying the peers responsible of each cluster is bounded by the overlay routing cost *log(Np)* (where *Np* is the number of peers).

4.2 Effect of Transformation Function on Query Complexity

In this subsection the complexity of a geographic query is estimated analytically. The result should indicate the implementation parameters that need to be optimized in order to model the earth surface as a whole then, more specifically, an urban area, and finally some heterogeneous wireless cells. Recall that a range query is carried out as follows:

- First the clusters within the given query box need to be identified
- A *get(key)* message is sent for to reach the peer responsible of the first key within that cluster iteratively
- Hopping between peers covering the cluster follows recursively

Overhead estimation of the search process:

To illustrate that, Figure 6 shows how the search for a single cluster is done (e.g. cluster 1: [6,11]). In total, *O(log(Np)+M)* messages are required.

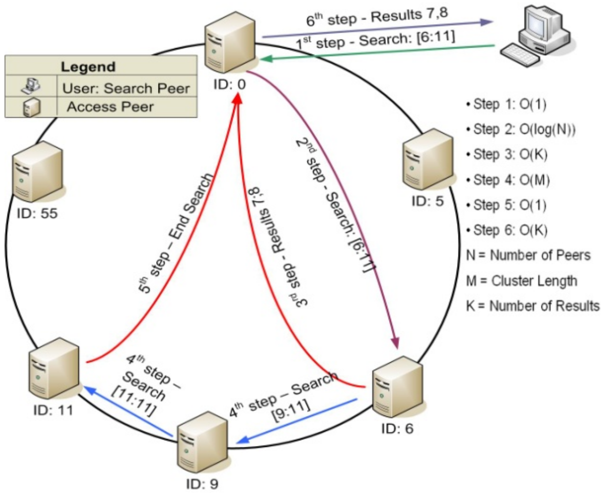

Fig. 5. Search for cluster 1 (from ID 6 to ID 11) shown in Figure 5

Table 1. List of variables used in the asymptotic study

Variables	Definition
$\gamma = k+n$	Binary index of a side distance of a 2-dimensional grid space (2^{k+n} unit for the maximum x-distance and 2^{k+n} unit for the maximum y-distance)
K	Binary index of a square search window, i.e. an area of $2^k \times 2^k$
H^2_{k+n}	A 2-dimensional Hilbert Curve filling a Geographic grid space with an area $2^{k+n} \times 2^{k+n}$
$N_2(k, k+n)$	The average number of clusters within a query box of size $2^k \times 2^k$ in a $2^{k+n} \times 2^{k+n}$ region filled by a H^2_{k+n} Hilbert Curve
M	The number of peers covering a single cluster (worse case)
M'	The average number of peers storing keys for a single cluster (simulated partition of the key space per peer)
Np	Total number of peers taking part in the Chord network
Np'	Number of peers storing data items for a portion of the grid space, e.g. a city

The average number of clusters within a query box of size $2^k \times 2^k$ in a $2^{k+n} \times 2^{k+n}$ region, covered by a H^2_{k+n} Hilbert curve is given as $N_2(k, k+n)$ 8:

$$N_2(k,k+n) = \frac{(2^n - 1)^2 2^{3k} + (2^n - 1)2^{2k} + 2^n}{(2^{k+n} - 2^k + 1)^2} \qquad \textbf{(1)}$$

The value of *M* indicates the fragmentation of a cluster among neighboring peers (two peers in Figure 6). For worst case estimation, we assume that each key is located at a different peer. The value of *M* can then be estimated as the length of each cluster.

$$M(k,k+n) = \frac{\mathit{number\ of\ keys\ in\ a\ query\ box}}{\mathit{avg\ number\ of\ clusters\ within\ that\ box}} - \mathit{first\ hop}$$

Or:

$$M(k,k+n) = \frac{2^k \times 2^k}{N_2(k,k+n)} - 1$$

The resulting complexity of the query box of size $2^k \times 2^k$ in space by a 2-dimentional Hilbert curve of degree ($\gamma = k+n$) is given in Equation (2).

$$O(N_2(k,k+n) \cdot \log_2(N_p) + 2^{2k} - N_2(k,k+n)) \qquad \textbf{(2)}$$

Analytic optimization of query overhead:

Figure 7 shows the effect of choosing the right Hilbert curve degree on the overhead of a range query. The earth surface could be modeled with different cell-sizes translating in different Hilbert curve degrees (*k+n*). The cases shown in Figure 7 are those of *(k+n) = 19, 18, 17, 16,* resulting in a Chord key space *of 38, 36, 34, and 32* bits respectively.

The cell size of each grid cell is given in the figure. The query box (see Table 2) is varied from *k = 0* to *k = 12,* resulting in a different geographic range depending on the grid size used. The more granular the Hilbert curve is, that the larger the overhead becomes. This is partly due to the fact that each object (in our case a wireless cell) of let say *500m* × *500m* size requires *2500* Chord keys when using $\gamma = k+n = 19$, and only *one* Chord key when using $\gamma = k+n = 15$. The problem with $\gamma = 15$, though, is that a smaller cell of let say *50m* × *50m* will be encoded as well in a *500m* × *500m* cell adding a significant loss of information about the stored data in the Chord ring.

The masking of the peer's ID reflects the fractal nature of the Hilbert curve. In the modelled space, the earth surface is covered by a $2^{19} \times 2^{19}$ grid, which covers the earth surface with a grid cells of a little more than $2^0 \times 2^0$ $1m^2$ (taking the earth surface to be *300 000 km*^{2}).

The effect of using a different grid cell size is shown in the Figure 7. This also corresponds mathematically to a different degree Hilbert curve and therefore to a different number of bits for the ID space (see Table 2).

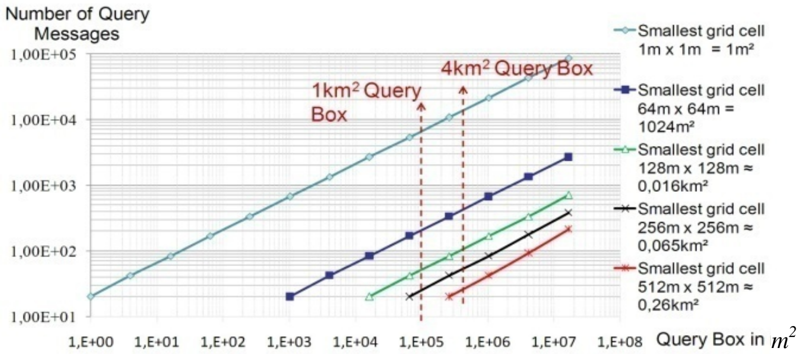

Fig. 6. Asymptotic varying query box for different Hilbert granularities

Table 2. Analytic evaluation of complexity for different Hilbert curve degrees

Parameter	Geographic Interpretation	Chord Interpretation
Earth Surface $2^{k+n} \times 2^{k+n}$	300 000 km^2	Total modeled surface
$2^{k+n} \times 2^{k+n}$	Number of cells in a grid covering the earth surface the size of each cell is *cell_size* = $\frac{3\times 10^{11}}{2^{2(k+n)}} m^2$	Hilbert Curve H^2_{k+n} the Chord key length is *(2·(k+n))-bit long*
$Np = 2^d$	$2^{2(k+n)}$ grid cells through 2^d peers	Each peer manages $2^{2(k+n)-m}$ possible keys, where *2·(k+n)>m*
m = 20	The surface covered by each peer is $\frac{3\times 10^{11}}{2^{20}} \approx 3\times 10^9 m^2$	At least a H^2_{10} or each peer has a 20-bit long Chord ID, the remaining *(2·(k+n)–20) bits* could be masked
Query Box $2^k \times 2^k$	Objects: Depending the modeled earth grid, each query box covers 2^{2k} × cell_size	Each query box requires $N_2(k, k+n)$ searches
Query Box $2^k \times 2^k$	Peers: number of involved access peers in the search is $\left\lceil 2^{2k} / 2^{2(k+n)-m} \right\rceil$	Each cluster requires *M'* recursive hops

Masking used in Figure 8 allows us to address peers with 20-bits instead of the full 38-bit full length IDs. Therefore the partition of the modeled clusters could be optimized. The actual length of a cluster becomes *M' (*see Table 2*)* where:

$$M' = \begin{cases} 1 & , in\ case\ \dfrac{2^{2k}}{N_2(k,k+n)} - 1 < 2^m \\ \dfrac{2^{2k} - N_2(k,k+n)}{N_2(k,k+n)} + 1 & , otherwise \end{cases}$$

Where $2^{2(k+n)-m}$ is the minimal distance between each two Chord nodes. In our example we took *m=20*.

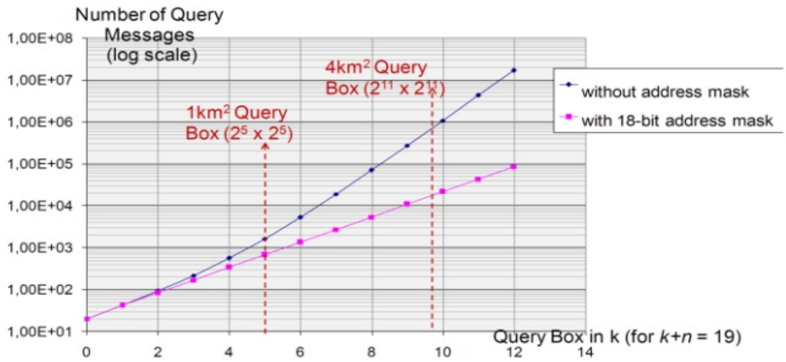

Fig. 7. Search complexity worst case scenario ($k+n$ = 19, m=20), 18-bit peer ID masking

5. Simulation Scenarios

Java-based discrete event simulation is used to validate the analytic model. Only the overlay behavior is targeted. The user mobility and query needs differ according to the assumed scenarios. Lookup strategies are chosen according to the modeled user behavior and the required queries are generated to mainly evaluate the system.

5.1 Network Model

The simulation model assumptions are summarized in Table 3:

Table 3. Network Simulation Model

Elements	Values
Network model	Chord nodes are simulated at the overlay level, i.e., fully reliable nodes and no packetization or delays
Overlay nodes	Both access and search peers are simulated. Chord is implemented. Node IDs are obtained using the Hilbert curve partition of the key space
Simulated object space	Using a fixed grid cell size of $16 \times 16\ m^2$, i.e., 2^{38-8} possible keys or 30-bit key space (*k+n=15*)

Urban area	268 km^2 large, where access networks and users are located, e.g. Passau in Germany is about 69km^2 large
Peer distribution	2^{10+p} of which, 2^p responsible of the 268 km^2 city and the remaining 2^{10} for the rest of the earth
Chord IDs per peer	Each peer within the urban area manages $2^{2(k+n)-m}$ Chord keys and covers a geographic zone of ($16^2\ m^2\ * 2^{30-p}$)
p	10, 9, 8, 6, and 4
Peer ID mask	Masking 20-bit peer IDs[2]

5.2 Object and Query Model

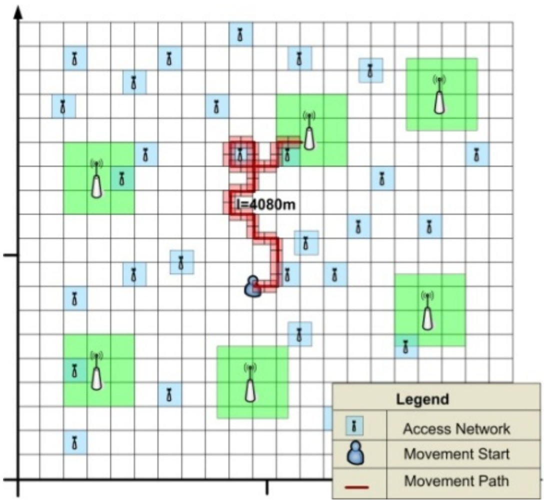

Fig. 8. A portion of the simulated environment with example query boxes in red

Given the geographic area of a medium-sized city, heterogeneous wireless cells are modeled as randomly distributed cells. A wireless cell of radius r is fitted inside a square grid cell of size $2r$ as already explained in Section 4. Similar to 15, 50 wireless cells are distributed randomly for each $1km^2$. That makes about 13400 access networks for the whole $268km^2$ modeled city (See Table 3).

Since wireless cells are heterogeneous in nature, we consider different cell sizes. Macro cells whose radius is up to 128m result in grid area of $256m \times 256m \equiv 16m{\cdot}2^s \times 16m{\cdot}\ 2^s$, *where* $2^{2\cdot s}$ represents the number of keys required to encode this cell in the Chord space, where "*s*" is varied uniformly between 1 and 4.

A graphical representation of part of the urban city is shown in Figure 9. Further to the object model, the query is also modeled according to some movement pattern assumptions.

- We assume that the user follows a random walk through a Manhattan type of street model.

---

[2] Since Chord uses a fixed ID space, each peer gets a 20-bit long ID with the remaining 10 least significant bits set to zero

- The Manhattan grid street system assumes a junction each *256m* a randomly selected direction is taken at each junction with probabilities: *0.5* for continuing in the same direction and probability *0.25* for turning either right or left.
- The starting point of the movement is fixed as well as the travelled distance to *4080m*.
- The velocity of the movement restricts the scope and maximum size of the query box, since the movement pattern can change by each street junction, i.e. every $(\tau = \frac{256m}{velocity})$ seconds. The resulting query boxes could be said to have the following properties.
- Assuming smaller query boxes as the full length of the traversed path, query boxes are emitted by the user recursively as he/she progresses along the movement path.
- A query box cannot be generated as long as the user has not moved out the area where his previous query has be generated, this excludes for slow users to generate large queries to frequently.
- A user moving with a given linear velocity crosses each single Chord key every *16m*.
- Each side of a query box covers a distance of $2^k \cdot 16m$. Given the user requires to generate queries periodically with a period $\Delta\tau$ then the query box could be defined as follows:
- $k = \log_2 \frac{velocity \times \Delta\tau}{16m}$, where k is a positive integer.

Assuming three different speeds (*2m/sec, 8m/sec,* and *16m/sec*) corresponding to (*7.2km/h, 28.8km/h,* and *57.6km/h*) respectively, and different periods for generating queries $\Delta\tau = 1, 2, 4, 8, 16, 32, 64$ sec , the resulting minimum query size is given in Table 4. The resulting number of recursive query boxes for each k and each speed is listed in Table 5.

Table 4. The possible exponent k used for the query box

-	$\Delta\tau$ =1	$\Delta\tau$ =2	$\Delta\tau$ =4	$\Delta\tau$ =8	$\Delta\tau$ = 16	$\Delta\tau$ = 32	$\Delta\tau$ = 64
Velocity = 2m/sec	-	-	-	0	1	2	3
Velocity = 8m/sec	-	0	1	2	3	4	5
Velocity = 16m/sec	0	1	2	3	4	5	6

Table 5. Number of required query boxes for the length 4080m

-	$k = 0$	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$
Number of query boxes Q for *4080m*	255	128	64	32	16	8	4

5.3 Measurements

Measurements take both message counts of both query messages and those contained in the responses. The exact message size is not of interest here. It is simply assumed to be of several bytes (or *kbits*). This is an implementation issue and depends on the format of the messages.

The messages sent back as part of a response are also measured as both message units (where each unit represents an additional Chord key sent back).

It is worth noting that a peer does not send a message unit for each key found, but rather constructs an XML description file, for instance, with a list of found keys. The size of the XML file however varies for each scenario.

5.4 Simulation Results

Search Overhead vs. Query Results

For this purpose, the network topology and object models are those found in Table 3, Table 4, and in Table 5. The number of peers covering the urban zone is initially set to 2^{10}, i.e. $m=10$ in Table 3. For the same distance of *4080m,* the number of query boxes needed are given in Table 4, whereas the frequency of the queries is $\Delta\tau$.

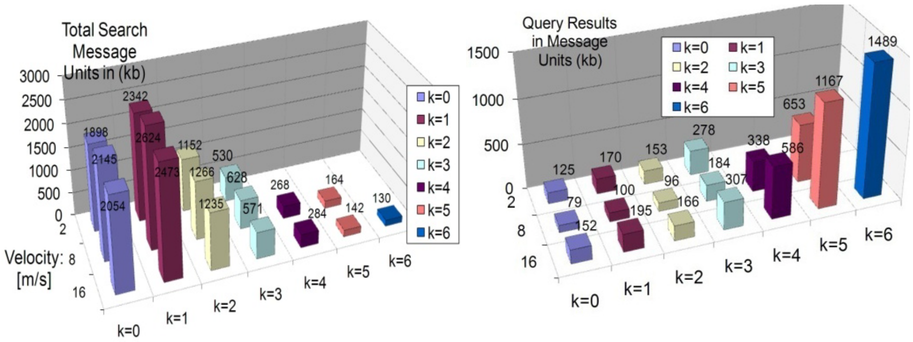

Fig. 9. Simulation results, for 4080m random path for test pair (*k,velocity)*:
(a) search messages overhead; (b) response messages overhead

The involved overlay messages including all the *get(key)* calls needed to cover each query box is measured. According to the Hilbert transformation of both the user's position, and the clusters included in each box, the final number of messages are summed up together in Figure 10 (a). For $k = 0$, the shown results in Figure 10 (a) are those of an exact match query occurring along the movement path and therefore repeating *255* times (see Table 5). The first range query occurs by $k = 1$ resulting in a large amount of messages.

It could be concluded that the larger the query box is that the fewer the number of messages needed to cover the same range (Figure 10-a). For each scenario a different random path has been selected indicating the slight difference that exists for each speed. Recall dependent on the size of the query box a Chord search is started for each cluster inside that box. Due to the random direction of our random path, there might be some slight differences in the number of clusters that result for each path.

The amount of recorded key responses (message units), shown in Figure 10-b, increases with increasing *k*. The case for *k=0* shows that the number of found keys vary according to the path taken as well as according to the object distribution. Due to the randomness of the two parameters, some difference could be noticed for each speed.

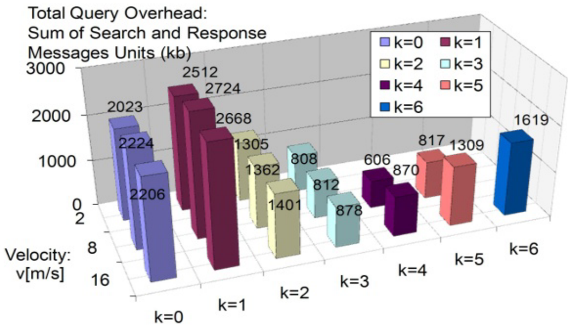

Fig. 10. Sum of search and response message overhead for each test pair *(k,velocity)*

The summation of both query messages and response message units - (see Figure 11) - shows the effect the selected query box for the selected context. The ideal query size would be in that case $k = 3$ or $k = 4$. But since $k = 4$ does not suit a slow moving user with a random walk as defined in Table 4, it could be concluded that $k=3$ is the more suitable query box size extracted from all three movement velocities.

The velocity of the movement shows little effect on the message count itself, but it could be said that the overall throughput cost of the range query could be calculated as follows:

$$Query\ Throughput = \frac{query\ messages\ count \times velocity}{distance}$$

This is an indication of the scalability of the whole approach. Other mechanisms such as Caching (at the search peers for instance) could be used to re-use similar query results generated from different users. Assuming that queries are initiated simultaneously, originating from different peers, the asynchronous nature of the query definition plays a major role in limiting the traffic. A slow user generates queries at a totally different frequency to a fast user. The fast user could generate larger queries to limit the frequency of the messages sent but then would have to tolerate a given error in the response messages.

A Query Error Measure

The error of a query is given as a function of the exact match query results (i.e., with $k = 0$) as follows:

$$Query\ Error_k = \frac{query\ response\ count\ (k) - query\ response\ count\ (for\ k = 0)}{query\ response\ count\ (for\ k = 0)}$$

The query error mostly has a computational cost, since the user has to filter out the messages that are sent back. For the case of *k=3* which in our scenario has produced the ideal amount of traffic messages, between 73% and 108% is noted (depending on the path selected). The trend of the query error follows that of the number of obtained results as shown in Figure 10-b. The query error is plotted for each *(k,velocity)* pair in Figure 12.

$$Query\ Error = \frac{query\ response\ messages - number\ of\ responses\ for\ (k=0)}{number\ of\ responses\ for\ (k=0)}$$

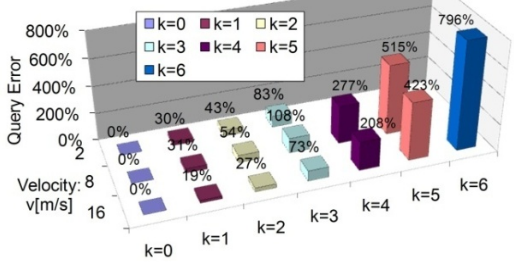

Fig. 11. Query error *(%)*

5.5 Comparing the Simulation Results with the Numerical Ones

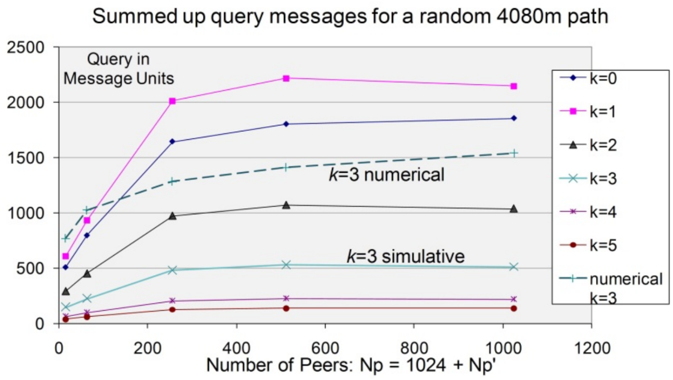

Fig. 12. The effect of number of urban peers (*Np'*)

In Figure 13, the number of peers which are managing the same urban zone is varied. The behaviour of the query message cost is compared. The query responses depend on the objects stored in the system and in this case this has been kept the same i.e. 13400 wireless cells managed by our urban peers.

The smallest number of urban peers (*Np'= 16*), is the most centralized scenario. The most distributed scenario simulates *Np' = 1024* as given in Table **3**. This indicated by parameter *"p = 4, 6, 8, 9, 10" corresponding to Np' = 16, 64, 256, 512, 1024).*

For a smaller query boxes ($0 \leq k \leq 2$), the query overhead is mostly affected by the number of the peers. For larger query boxes *(*$3 \leq k \leq 4$*)* the effect of the number of peers is minimal. Now recall our complexity equation (Equation (2) in Subsection 4.2) $O(N_2(k,k+n)\cdot\log_2(N_p)+2^{2k}-N_2(k,k+n))$. The overhead cost is repeated for each single query box that covers the movement path. In our simulated scenario, this represents a multiplication factor Q given in Table **5**. The cost complexity is then transformed to:

$$O(Q\cdot[N_2(k,k+n)\cdot\log_2(N_p)+2^{2k}-N_2(k,k+n)])$$

Where:

$$Q=\frac{full\ length\ movement\ path}{grid\ cell\ size\times 2^k}=\frac{4080m}{16m\times 2^k}$$

Furthermore, the number of clusters $N_2(k,k+n)$ (see Equation (1), Subsection 4.1) decreases with increasing size of the query window k. This also explains the influence of the logarithmic factor (*log(Np)*) on the overhead cost. For larger k the number of clusters decreases, making the influence of $N_2(k,k+n)\times\log(Np)$ part of the equation smaller. For smaller k this part of the equation leads to a significant increase.

Another result – *explained but not plotted here due to lack of space* – is the number of overlay hops required to reach each cluster (i.e. log (N) in Chord). This has been measured to be around 0.5·log(*Np'*)[3] instead of log(*Np*), where *Np= 1024+Np'* (see Figure 13). The use of *Np'* could be explained by the fact that the search peer is close (on the overlay level) to the queried range. This differs to a scenario where a location-based query is started from Europe, for instance, and destined at the US (e.g. "*from Passau – Europe – find all green spaces in Chicago – US"*). The average routing cost is then given in Equation (3).

$$Average\ Cost=Q\cdot[N_2(k,k+n)\cdot 1/2\cdot\log_2(N_p')+2^{2k}-N_2(k,k+n)]) \qquad \textbf{(3)}$$

Based on Equation (3), the average numerical number of query messages (shown by the dashed line for *k=3)* is plotted in Figure 13. Equation (2) assumes that each key is placed at a distinct peer. Once the query reaches the cluster, *M* hops are needed to cover the cluster length (described by variable *M* in Table 1). In the simulation M' turns out to be at most one additional hop. This tendency is shown by the simulative results for *(k=3)* when compared with the numerical result (Equation 3). Both results follow the same growth tendency. The logarithmic growth (i.e., $log_2(2\cdot Np')=log_2(Np')+1$) explains the flattening of the cost after *Np'= 512* in all cases. It is hard to predict, numerically, the splitting of a cluster between peers. In the case of *k=1,* the clusters' partition among peers has lead to a slight decrease in the search cost (from *Np' = 512* to *Np'= 1024*).

[3] In Chord the average number of hops is given as 0.5*log(N), where N is the number of peers14

6. Conclusions

The proposed context-aware handover relies on application layer overlay to access information stored by heterogeneous data bases. Based on the clustering properties of geographic information, the Hilbert curve is used to obtain a transformation function between the object storage attribute (geographic zones) and their allocated ID in the overlay system. The fractal nature of the Hilbert curve is exploited to address overlay nodes while preserving the clustered structure of the managed data. Numerical analysis of the query problem is used to further analyse the complexity of the sought system. A simulation study, then, investigates the performance of the range queries and compares the results to the expected numerical cost estimation.

The proposed solution is shown to be affected by very local decisions. For instance the size of the query could be optimized locally, and the effect of the Chord complexity only takes the local urban context into account.

As a future work, further scalability studies of the system will be investigated as well as applicability of the design path to other distributed management systems.

Acknowledgements

This work has been supported by the EURO-FGI -Network of Excellence, European Commission grant IST 028022 and specifically the subprojects VNets and MMSOS. The authors would like to thank Andreas Berl for providing insightful comments to the ideas presented in this paper, and Ivan Dedinski who provided valuable support for the simulation work.

References

1. Akyildiz, I. F., Wang, X., Wang, W.: Wireless Mesh Networks: A Survey. Computer Networks Journal, Elsevier. 47(4), pp. 445—487 (March 2005)
2. Akyldiz, I. F., Xie, J., Mohanty, S.: A Survey of Mobility Management in Next Generation all-IP based Wireless Systems. IEEE Wireless Communications. 11(4), pp. 16-- 28 (August 2004)
3. Balasubramaniam, S., Indulska, J.: Vertical Handover Supporting Pervasive Computing in Future Wireless Networks. Computer Communications. 27(8), pp. 708-719 (May 2004)
4. Datta, A., Hauswirth, M., John, R., Schmidt, R., Aberer, K.: Range Queries in Trie-Structured Overlays. In: Proc. of Fifth Peer-to-Peer Computing Conference, pp. 57-66. IEEE Press, Konstanz, Germany (31 Aug.-2 Sept. 2005)
5. Houyou, A. M., De Meer, H., Esterhazy, M.: P2P-based Mobility Management for Heterogeneous Wireless Networks and Mesh Networks. In Cesana, M., Fratta, L. (eds) Wireless Systems and Network Architectures in Next Generation Internet, Menaggio Como, Italy, July 13 - 15, 2005. LNCS, vol. 3883, pp. 226—241. Springer, Heidelberg (2006)
6. Knoll, M., Weis, T.: Optimizing Locality for Self-organizing Context-Based Systems. De Meer, H., Sterbenz, J. P. G. (eds), Self-Organizing Systems, First International Workshop, IWSOS 2006, Passau, Germany, September 18-20, 2006. LNCS, vol. 4124, pp. 62—73, Spinger, Heidelberg (2006)
7. Maymounkov, P., Mazieres, D.: Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In: Goos, G., Hartmanis, J., van Leeuwen, J. (eds) 1st International

Workshop on Peer-to-Peer Systems (IPTPS), Cambridge, MA, USA, March 2002, LNCS, vol. 2429, pp. 53—65. Springer, Heidelberg (2002)
8. Moon, B., Jagadish, H. V., Faloutsos, C., Saltz, J. H.: Analysis of the Clustering Properties of the Hilbert Space-Filling Curve. In IEEE Transactions on Knowledge and Data Engineering. 13(1), pp. 124—141, IEEE Press, (2001)
9. Ratnasamy, S., Francis, P., Handley, M., Karp, R., and Schenker, S.: A scalable content-addressable network. In SIGCOMM'01: Proceedings of the 2001 conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, pp. 161—172. ACM Press, New York (2001)
10. Sahin, O. D., Gupta, A., Agrawal, D., El Abbadi, A.: Query Processing Over Peer-to-Peer Data Sharing Systems. Technical Report UCSB/CSD-2002-28, University of California at Santa Barbara, 2002
11. Schiller, J., Voisard, A.: Location-Based Services. Morgen Kaufmann/Elsevier, San Fransisco, CA, USA, (2004)
12. Sethom, K., Afifi, H., Pujolle, G.: Palma: A P2P based Architecture for Location Management. In 7th IFIP International Conference on Mobile and Wireless Communications Networks (MWCN 2005), Marrakech, Morocco, September 19-21 2005
13. Stemm, M., Katz, R. H.: Vertical Handoffs in Wireless Overlay Networks. In Mobile Networks and Applications. 3 (4), pp 335-- 350. Springer, Netherlands (1998)
14. Stoica, I., Morris, R., Liben-Nowell, D., Karger, D. R., Frans Kaashoek, M., Dabek, F., Balakrishnan, H.: Chord: a Scalable Peer-to-Peer Lookup Protocol for Internet Applications. IEEE/ACM Transactions on Networking. 11(1), pp 17—32. (2003)
15. Zdarsky, F. A., Martinovic, I., Schmitt, J. B.: The Case for Virtualized Wireless Access Networks, De Meer, H., Sterbenz, J. P. G. (eds), Self-Organizing Systems, First International Workshop, IWSOS 2006, Passau, Germany, September 18-20, 2006. LNCS, vol. 4124, pp. 90—104, Spinger, Heidelberg (2006)
16. Zhuang, S., Lai, K., Stoica, I., Katz, R., Shenker, S.: Host Mobility Using an Internet Indirection Infrastructure. In Wireless Networks. 11(6), pp 741—756. (2005)